0-315-01202-1

National Library of Canada | Bibliothèque nationale du Canada

Canadian Theses Division | Division des thèses canadiennes

Ottawa, Canada K1A 0N4

49060

PERMISSION TO MICROFILM — AUTORISATION DE MICROFILMER

• Please print or type — Écrire en lettres moulées ou dactylographier

Full Name of Author — Nom complet de l'auteur

Gültekin Özsoyoğlu

Date of Birth — Date de naissance

15-9-1951

Country of Birth — Lieu de naissance

TURKEY

Permanent Address — Résidence fixe

Alper Apt. 8/11
Alper Sok.
Gaukaya /ANKARA
TURKEY

Title of Thesis — Titre de la thèse

Secure Statistical Database Design

University — Université

University of Alberta

Degree for which thesis was presented — Grade pour lequel cette thèse fut présentée

Ph.D.

Year this degree conferred — Année d'obtention de ce grade

1980

Name of Supervisor — Nom du directeur de thèse

Francis Y. Chin

Permission is hereby granted to the NATIONAL LIBRARY OF CANADA to microfilm this thesis and to lend or sell copies of the film.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.

L'autorisation est, par la présente, accordée à la BIBLIOTHÈQUE NATIONALE DU CANADA de microfilmer cette thèse et de prêter ou de vendre des exemplaires du film.

L'auteur se réserve les autres droits de publication; ni la thèse ni de longs extraits de celle-ci ne doivent être imprimés ou autrement reproduits sans l'autorisation écrite de l'auteur.

July 31, 1980

Date

Signature

National Library of Canada
Collections Development Branch

Canadian Theses on
Microfiche Service

Bibliothèque nationale du Canada
Direction du développement des collections

Service des thèses canadiennes
sur microfiche

NOTICE

The quality of this microfiche is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us a poor photocopy.

Previously copyrighted materials (journal articles, published tests, etc.) are not filmed.

Reproduction in full or in part of this film is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30. Please read the authorization forms which accompany this thesis.

THIS DISSERTATION
HAS BEEN MICROFILMED
EXACTLY AS RECEIVED

AVIS

La qualité de cette microfiche dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de mauvaise qualité.

Les documents qui font déjà l'objet d'un droit d'auteur (articles de revue, examens publiés, etc.) ne sont pas microfilmés.

La reproduction, même partielle, de ce microfilm est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30. Veuillez prendre connaissance des formules d'autorisation qui accompagnent cette thèse.

LA THÈSE A ÉTÉ
MICROFILMÉE TELLE QUE
NOUS L'AVONS REÇUE

Ottawa, Canada
K1A 0N4

NL-339 (Rev. 8/80)

THE UNIVERSITY OF ALBERTA

SECURE STATISTICAL DATABASE DESIGN

by

© GÜLTEKİN ÖZSOYOĞLU

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTING SCIENCE

EDMONTON ALBERTA CANADA

FALL 1980

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research, for acceptance, a thesis entitled "SECURE STATISTICAL DATABASE DESIGN", submitted by Gultekin Ozsoyoglu in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Francis Y. Chin (supervisor)

Date July 31, 1980

University of Alberta

To my wife, Meral,

and my parents, Aliye and Şükrü Özsoyoğlu

ABSTRACT

Statistical databases provide statistical information to user queries. The security problem for a statistical database is to limit the use of the database so that no sequence of queries is sufficient to deduce confidential information about any individual. In this thesis, the security problem of statistical databases is investigated in the context of a statistical database (SDB) design. New results involving a comprehensive secure SDB design are described.

A partitioning model of the SDB is discussed in order to be used as a tool in the SDB design. Primitive change operations are allowed in the model, and the conditions are derived to prevent compromise. Variations of the partitioning model which use either rounding, or data perturbation or both are introduced to remove some of the assumptions made in the partitioning model.

The importance of semantic meaningfulness of users' queries is stressed. It is argued that it will enhance the security by not permitting malicious users to form meaningless queries in order to use their responses in combinatorial formulas for compromise. Within the context of a formal framework, an SDB design using security constraints at the conceptual data model level is proposed. Three different structural, semantic and redundant data models are

investigated and the D-A model [Smith and Smith, 1977] is chosen as a conceptual data model of the SDB. The population concept is utilized to identify semantically well-defined objects about which statistical information is revealed to users. For this purpose, the Population Definition Construct (PDC) is introduced for each population in the conceptual model.

It is argued that, for complete protection, users' additional knowledge should be maintained and kept up-to-date. Users' additional knowledge may take the form of general rules and explicit facts. The SDB design proposed herein maintains only the users' knowledge of protected property values of individuals in the SDB using the User Knowledge Construct (UKC).

In order to keep the PDCs and UKCs up-to-date, to enforce the security constraints and to help the DBA in security-related decision problems, the constraint enforcer and checker (CEC) is proposed. The CEC, UKCs and PDCs comprise the Statistical Security Management Facility (SSMF). Implementation issues of the SSMF are briefly discussed.

Different types of inferences by users are identified, and possible security constraints for different types of statistical queries are investigated. It is demonstrated that, usually, simple security constraints can be defined to

protect the SDB from compromise.

Extensions to the SDB design are described which includes a Question-Answering System, a security kernel and a set of security-related high level commands for handling the changes in the SDB.

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to my supervisor, Dr. Francis Y. Chin, for his guidance and assistance during the course of this project. For two years, he undertook the task of reading (long!) weekly reports, and discussing the subject and otherwise for several hours with me.

I am grateful for the careful reading and comments provided by the members of my examination committee, Drs. Len Schubert, Dorothy Denning, Mary McLeish, Tony Marsland and Dale Bent.

Everybody in the Computing Science Department have contributed to making my wife's and my four year stay here a very enjoyable experience. Jim Lamont introduced us to the Canadian Rockies and Wilderness; for us, a life-long, beautiful memory.

I gratefully acknowledge the financial support of the Computing Science department in the form of teaching and research assistantships.

Meral, my wife, has been a constant source of inspiration, and deserves special thanks for everything she has done for me. Indeed, without her, this thesis would never have been completed.

TABLE OF CONTENTS

CHAPTER		PAGE
1:	INTRODUCTION	1
1.1	Computer Security	1
1.2	Statistical Database Security	3
1.3	Outline of the Thesis	6
2:	SDB MODELS AND PROTECTION SCHEMES	8
2.1	SDB Models	8
2.2	Protection Schemes	10
3:	PARTITIONING MODELS WITH DATA AND OUTPUT PERTURBATION	21
3.1	A Partitioning Model	21
3.1.1	An Information Graph	23
3.1.2	Properties of the Information Graph and a Security Result	26
3.2	Partitioning Models with Data and Output Perturbation	32
3.2.1	A Partitioning Model with Data Perturbation	32
3.2.2	A Partitioning Model with Rounding	34
3.2.3	A Partitioning Model with Rounding and Data Perturbation	38
3.3	Security Measures Related to Information Graph	41
4:	STATISTICAL DATABASE DESIGN	43
4.1	Introduction	43

4.2 Statistical Database Design 46

4.2.1 Constituents of Statistical Information. 49

4.2.2 Formal Definitions of SDB and the Security Problem 50

4.2.3 Conceptual Data Models for SDB Design... 58

4.2.4 Statistical Information Related to Each Population 71

4.2.5 Security Constraints 74

4.3 A Statistical Security Management Facility 77

4.3.1 Implementation Considerations 86

4.4 Protection Requirements for Different Statistical Queries 93

4.4.1 COUNT Queries 94

4.4.2 SUM and COUNT Queries 96

4.4.3 MEDIAN and COUNT Queries 97

5: EXTENSIONS FOR A SECURE SDB DESIGN 103

5.1 Introduction 103

5.1.1 Inferred Knowledge and a Question-Answering System 103

5.1.2 Changes and Adaptability of the Conceptual Model 109

5.1.3 Security Kernel for the SDB 111

5.2 The Extended SDB Design 112

5.2.1 The Protection Data 114

5.2.2 The (Extended) SSMF and the QAS 115

5.2.3 Conceptual Model Modification and Security-Related Commands 117

5.3 Processing Queries and Commands 125

5.4 Discussion 134

6: CONCLUSIONS 137

BIBLIOGRAPHY 142

APPENDIX A: Proof of Lemma 3.1 148

APPENDIX B: Usage of Dummy Records in the Partitioning Model 151

LIST OF TABLES

Table	Description	Page
3.1	The Partitioned Database Model	31

LIST OF FIGURES

Figure	Description	Page
2.1	Partitioned Database of Customer Accounts ...	15
3.1	Information Graph of Example 3.1	26
4.1	Classification of Facts and Inference Rules Relevant to the Application	57
4.2	Decomposition of the Generic Object Computer Scientist	64
4.3	Decomposition of the Generic Object Computer Scientist with PhD	64
4.4	Generalization Hierarchy for Computer Scientist	65
4.5	Database of Employees, Assignments and Projects	66
4.6	A Characteristic Tree	69
4.7	The PDC of Programmer	79
4.8	The UKC of User Group u for the Generic Hierarchy Described in Figure 4.4	82
4.9	Statistical Database Model	85
5.1	The Create Population Command	119
5.2	The Decompose Command	122
5.3	The Create Property Command	122
5.4	Retrieval Operation for Statistical queries .	126
5.5	User- or DBA-Requested change operation	130
5.6	DBA-Requested conceptual model change operation	133
5.7	Execution of a DBA-Requested Security-Related Command	135

CHAPTER 1

INTRODUCTION

1.1 Computer Security

Computer security has always been an important issue. However, due to the wide-spread usage of computers and large quantities of shared data, the issue has gained far greater importance.

Parker [Parker, 1976] reports that the median and the total known loss in the cases of computer abuse were around $500,000 and $100 million annually. These figures are expected to rise [Denning and Denning, 1979] unless countermeasures are taken. Goal of the computer security research is to devise safeguards to prevent possible computer abuse.

Definitions of Security, Privacy and Confidentiality are presented below [ACM, 1974]:

Security. Data security is the protection of data against accidential or intentional destruction, disclosure, or modification. Computer security refers to the technological safeguards and managerial procedures which can be applied to

computer hardware, programs, and data to assure that organizational assets and individual privacy are protected.

Privacy. Privacy is a concept which applies to an individual. It is the right of an individual to decide what information (s)he wishes to share with others and also what information (s)he is willing to accept from others.

Confidentiality. Confidentiality is a concept which applies to data. It is the status accorded to data which has been agreed upon between the person or organization furnishing the data and the organization receiving it and which describes the degree of protection to be provided.

Security mechanisms may be classified into two groups [Denning and Denning, 1979]. Internal security mechanisms control the operation of the computer system in four areas: access control to stored objects, information flow control between stored objects, encryption of confidential data transmitted on communications lines, and inference control of confidential data stored in statistical databases [Denning and Denning, 1979; Hsiao et al., 1978; Madnick, 1979]. External security mechanisms control operations outside the main computing system, examples are fire protection, personnel screening, etc. [Madnick, 1979; Shankar, 1977; Nielsen et al., 1976].

This thesis is concerned with the inference control of confidential data stored in statistical databases.

1.2 Statistical Database Security

A statistical database (SDB) has been defined as one which returns statistical information, such as frequency counts of records satisfying some given criteria, as opposed to a database which returns complete details of a record, for example name and address of an employee. Such statistical databases have wide applicability in medical research, health planning and political planning, to name just a few.

The security problem for a statistical database is to limit its use so that only statistical information is available and no sequence of queries is sufficient to derive confidential information about any individual. When such information is obtained the database[+] is said to be compromised (or disclosure has occurred). Notice that the protected information does not necessarily reside in the database.

In order to clarify the nature of the security problem, three examples are now presented. First consider an on-line system that gives information about the number of individuals having certain properties (i.e., COUNT information is revealed). The system tells the user that a total of three people have the following properties; age 39,

[+]In this thesis, the terms database and statistical database will be used interchangeably.

male, married, live in Edmonton and are lawyers. Suppose further that the user knows a particular lawyer having all of above characteristics and his data is included in the database. Now if the user enquires the number of people having all of the above properties and in addition with earnings over $50,000 a year, and gets an answer of 3, then the user u knows immediately that this particular lawyer earns over $50,000 a year.

As another example, consider an off-line system, e.g., a census publication office, that publishes tables of statistical information. Suppose a small county has six hardware stores and a city within the county has four of them. If retail sales are published for the county and for the city then each of the two out-of-town stores can determine the other's sales simply by taking differences between the published county and city figures.

For the third example consider an on-line database of employees of a company, in which salary ranges for employees are not protected, It is also known that every electrical engineer with salary < $20,000, at least 5 years working experience and a B.S. degree has had at least one "bad" rating from his manager. Suppose the user knows an electrical engineer with a B.S. degree and experience for more than 5 years in the company. If both queries, "number of electrical engineers with B.S. degree and at least 5 years experience" and "number of electrical engineers with

B.S. degree, at least 5 years experience and salary < $20,000" have the same answer then that particular electrical engineer has had at least one "bad" rating from his manager.

The first two examples illustrate two different kinds of applications, an on-line and off-line application while the third example illustrates the fact that the information deduced by the user may not be stored in the database. For the off-line application, statistical offices traditionally examine their publications carefully to ensure that there is no disclosure. However increasing demand for detailed information and possible use of computers for correlating several publications to disclose further information have prompted researchers to consider more strict security measures, such as data perturbation techniques [Hansen, 1971; Nargundkar and Saveland, 1972; Fellegi and Phillips, 1972]. In the case of on-line databases, instead of storing aggregate information, the database contains anonymous but individual records, and returns statistical summaries of those records which satisfy the specific characteristics given in the query. Changes to the database, such as insertions, deletions and updates, are allowed and responses to queries are expected to reflect the current status of the database.

1.3 Overview and Outline of the Thesis

In Chapter 2, an overview of the previous studies on SDB security is provided. Using a set of "goodness" criteria, proposed protection policies are discussed and evaluated.

A partitioning model for dynamic SDBs is investigated in Chapter 3. The information revealed to the users during the insertions, deletions and updates is characterized and it is shown that, under certain conditions, the model is secure. Data perturbation and rounding are proposed to remove some of the disadvantages of the model. Some security measures which help the DBA to assess how secure the SDB is at a certain time are defined.

Using a formal framework in Chapter 4, the design of an SDB which employs security constraints at the conceptual data model level is investigated. Three redundant, structured and semantic data models are analyzed for their suitability as a conceptual model of the SDB. In the SDB, information revealed to users is well-defined in the sense that it can at most be reduced to indivisible information involving a group of individuals. Any information involving few individuals (and therefore risking compromise) is recorded and kept for auditing purposes.

In Chapter 5, the design of SDB is extended with a Question-Answering System, a security kernel and a set of

security-related high level commands.

Finally, the overall significance of all the results and an overview of the motivation for the research is outlined. Outstanding problems and areas that require further investigation are indicated.

CHAPTER 2

SDB MODELS AND PROTECTION SCHEMES

Proposed protection policies for SDB models are discussed and evaluated using a set of "quality" criteria.

2.1 SDB Models

Below is an SDB model proposed by Denning [Denning, 1978; Denning et al.; 1979].

A **statistical database** can be viewed as a set of n records. Each record has k attribute (property) values corresponding to attributes $A_1, A_2, \ldots, A_k$, among which some are protected attributes. Values of the protected attributes for each record are confidential and only statistical summary information about these attributes is available. An example is a database of employees. Each record has attributes NAME, ADDRESS, SEX, AGE, POSITION, etc. and a protected attribute SALARY.

A **query** is some statistical function, e.g. MEAN, MEDIAN, etc., applied to some subset of the records in the database. Every query has a **characteristic expression** C

which is a logical expression using the logical operators, conjunction (&), disjunction (v) and negation (~). The set of records satisfying the characteristic expression C of a query is called the **query set**, S(C). For example, in the database of employees, C=((AGE<40)&(POSITION=programmer)) is a characteristic expression and its query set S(C) contains all the programmers under 40 years of age.

The most common statistical query types can be defined as follows:

COUNT(C)=|S(C)|, the size of S(C)

SUM(C,A_i)=sum of the i^{th} attribute values of those records in S(C), $1 \leq i \leq k$

AVERAGE(C,A_i)=average of the i^{th} attribute values of those records in S(C), $1 \leq i \leq k$

MAX(C,A_i)=the maximum of the i^{th} attribute values of those records in S(C), $1 \leq i \leq k$

MIN(C,A_i)=the minimum of the i^{th} attribute values of those records in S(C), $1 \leq i \leq k$

MEDIAN(C,A_i)=the median of the i^{th} attribute values of those records in S(C), $1 \leq i \leq k$

The database is **compromisable** if one can deduce from the responses of the queries some protected attribute values of records. Clearly this definition is less general than the one given in the introduction since although an attribute may not be confidential it may nevertheless be necessary to protect it if it leads to the derivation of some

confidential information not necessarily in the database.

In some other SDB models key-specified queries are used to describe query sets [Dobkin et al., 1979; DeMillo et al., 1978; Reiss, 1979a]. Binary k-bit keys are used to describe attribute values of records and the k-bit queries with Ø's, 1's and *'s (don't care) for the query sets [Kam and Ullman, 1977; Chin, 1978]. An m-response system suppresses answers to queries with the query set size less than m.

2.2 Protection Schemes

The proposed protection schemes may be classified into the following six categories:

1) controlling the size of the query set,
2) limiting excessive overlap between query sets,
3) partitioning the database,
4) output perturbation,
5) random sampling,
6) data distortion.

In general protection schemes impose restrictions on the system. In order to compare the "quality" of these schemes, the following factors will be considered:

(a) Effectiveness: restrictions should guarantee security to a reasonable extent. We will also discuss the effectiveness of restrictions under dynamic databases and users' knowledge about the real world that the database models.

(b) Feasibility: it is possible that some restrictions are sufficient to guarantee security but the system has no way to enforce them. The enforcement of restrictions should be feasible.

(c) Efficiency: the implementation should be efficient. Any scheme which is virtually impossible to implement or involves too much overhead, should be avoided.

(d) Richness: restrictions when applied should not conceal too much information. In other words, the database should still be rich enough to be useful for users.

Protection by Controlling the Size of the Query Set:

One of the earliest and most straightforward protection schemes is to suppress queries whose query set size is small [Hoffman and Miller, 1970; Hansen, 1971]. The examples given in Section 1.2 illustrates that the security of the database is endangered by allowing answers to queries with small query set size. In [Chin, 1978], an m-response system using k-bit binary keys to describe characteristics of records is introduced. It allows only SUM and COUNT queries and prohibits answering those queries whose query sets have less than m records. Necessary and sufficient conditions to quarantee the security of a 2-response system is given for a static database (i.e., no insertions, deletions and changes in the database). Unfortunately this result imposes too many restrictions on the system and limits the richness of the database. Moreover, as illustrated by the second example in

Section 1.2, since any two users of a particular kind in a 2-response system might easily know each other, it is a generally accepted practice to have an m-response system with $m \geq 3$. However, the properties of a m-response system for $m \geq 3$ are not well understood.

It can be shown that information can also be deduced if queries with a large query set size are answered [Denning et al., 1979]. Thus the system should limit queries with very small and very large query sets. Unfortunately, this protection scheme can easily be subverted by a device called tracker [Schlorer, 1975; Denning et al., 1979]. A tracker is some auxiliary characteristic expression which, when added to the original characteristic expression, produces an answerable query. The user then uses this answer with some others to deduce the answer for the original unanswerable query. This idea is further extended to double trackers and general trackers which are applied to more restricted ranges of answerable query set sizes.

For key-specified MEDIAN queries, the number of queries to compromise the SDB is lower bounded by $O(\log_2 k)$ queries and upper bounded by $O(\log_2^2 k)$ queries [DeMillo and Dobkin, 1979; Reiss, 1979].

The above results show that the technique of controlling the size of the query set is not effective (although feasible) and merely makes the intruder's job

harder.

Protection by Limiting Excessive Overlap Between Query Sets:

The protection scheme of limiting query set overlap assumes a static database of n records and a fixed query set size k, and inhibits the responses to queries whose query sets have too much overlap (say, more than r records) with the query sets of other answered queries. It is shown that, for SUM statistical queries, the smallest number of queries sufficient to compromise the SDB is lower bounded by S=(2k-(t+1))/r, where t is the number of records whose protected attribute values are known by the user [Reiss, 1979]. It is also shown that this bound, S, is optimum for r=1 and t=Ø,1 [Dobkin, 1979]. There are two problems with the protection by controlling query set overlaps. First, it may not be feasible since extensive set intersection checks are required. Second, since a previously answered query may inhibit the responses of several other more useful queries, this protection scheme may severely limit the richness of the database.

Threat monitoring is another proposed scheme that does not guarantee security but is claimed to provide a deterrent for intruders [Hoffman, 1977; Denning, 1978]. The system monitors queries that have been answered and tries to detect excessively active periods of use of a database and to detect instances of many successive and similar queries.

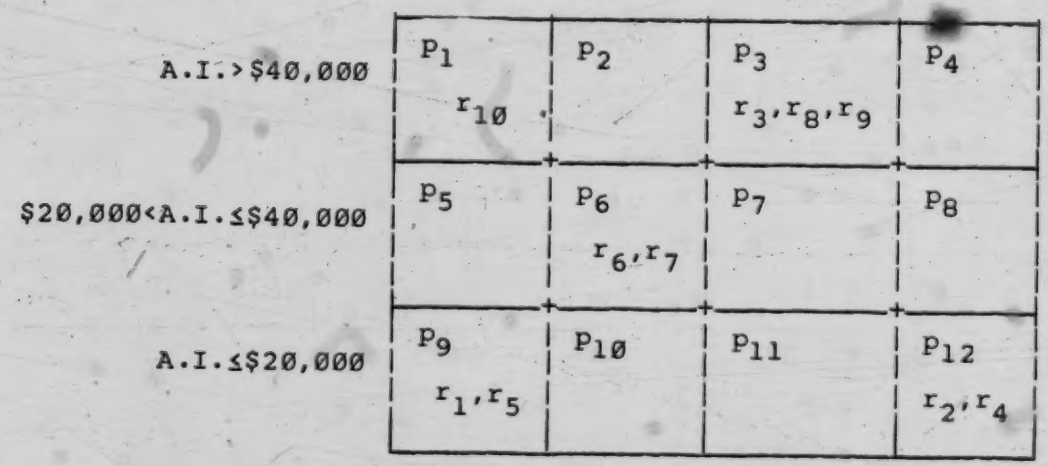
This scheme however can easily be made ineffective by masking queries [Schlorer, 1976].

Protection by Partitioning the Database:

In this protection scheme the whole database is partitioned into groups of records, each of which are in partitions defined over k attribute domains [Yu and Chin, 1977]. Each partition has either no records or at least u records with u>1. Queries are modified to report over partition boundaries. In other words, queries always involve pre-specified groups of records and never subsets of these groups. Thus records inside a group cannot be isolated by overlapping queries and only information concerning whole groups can be derived. However, if the database is dynamic, i.e. insertions, deletions and updates of records are allowed then each change in a group can be detected and the changed record can be disclosed. To illustrate various compromises, consider the example in Figure 2.1 which contains the customer accounts of a database and its partitioned model. Assume a query is presented for the total accounts of engineers with annual income > $38,000. This query is modified so that it returns the total accounts of the engineer customers with salary > $40,000 (i.e. partition p_1 in Figure 2.1). Since the records corresponding to engineer customers with income > $40,000 form a partition (p_1) and since there is only one customer record in that partition, the information in that record can be extracted.

Record No.	Name	Income	Profession	Account
r_1	N_1	$10,000	Engineer	$167
r_2	N_2	$15,000	Politician	$410
r_3	N_3	$45,000	Lawyer	$24
r_4	N_4	$20,000	Politician	$4210
r_5	N_5	$8,000	Engineer	$325
r_6	N_6	$30,000	Doctor	$12,000
r_7	N_7	$25,000	Doctor	$500
r_8	N_8	$50,000	Lawyer	$300
r_9	N_9	$60,000	Lawyer	$123
r_{10}	N_{10}	$90,000	Engineer	$20,000

(a) Records in the database



(b) Partitioned Model

Figure 2.1. Partitioned Database of Customer Accounts

Assume a new politician customer N_{11} with annual income \$12,000 has opened an account. If a user knows that the politician customer N_{11} has annual income $\leq$ \$20,000 and that N_{11} has recently opened an account in the bank then he can deduce the amount in N_{11}'s account by querying the partition p_{12} before and after the insertion of N_{11}'s record. Thus changes in the database must be processed in a controlled manner.

The partitioning model has the following deficiencies.

(1) Modifying queries to report over partition boundaries may conceal too much information unless the partition sizes are small, uniform and independent of the distributions of records' attributes. However, in order to avoid partitions with less than u records, variable size partitions are proposed [Yu and Chin, 1977], and unless u is small, this condition may create large partitions reducing the usefulness of the database. Also checking and modifying queries so that they report over partition boundaries, and accessing nonuniform partition boundaries may be costly.

(2) If the database is currently undergoing changes (i.e. insertions, deletions and updates) then each change has the danger of being detected and the values of those records involved may be disclosed. In [Yu and Chin, 1977], it is suggested that these changes be classified into three different groups (insertions, deletions and updates) and any changes in a partition should not be implemented until there

are t changes in the same group for that partition. This policy introduces an error in SUM queries, which is dependent on the values of records with changes and for large t, this error may reduce the usefulness of the statistical information. In the other extreme, however, if a change is implemented immediately after it is requested then by querying before and after the change, the information in the record involved with the change can be disclosed.

(3) In certain cases, users may already know some record values from sources other than the database. When this happens, some mechanisms are needed to decide which other record values are in danger. In other words, exact information revealed to users must be recorded and kept for auditing purposes.

(4) Some records may contain more sensitive information than the others and, while performing changes in these records, the database system may want to ensure that their disclosure is independent of the disclosure of other record values. In other words, the database system may want to exercise some control about the information revealed to users during changes in the database.

In Chapter 3, some proposals are made to remedy these deficiencies.

A variant of partitioning is grouping (or microaggregation) which is used in off-line applications such as census publications [Hansen,1971; Fellegi and

Phillips, 1974]. Records are grouped together and only aggregate statistical information is given. These two techniques, partitioning and grouping, have also the limitations of possible loss in the richness of the system, especially when the groups are ill-formed.

Protection by Output Perturbation:

All the protection methods discussed so far provide the user exact statistical information of the query set. Another protection scheme is to perturb the responses to the queries without losing too much of the meaningfulness of the information in the database. Rounding [Hansen, 1971; Nargundkar and Saveland, 1972; Achugbue and Chin, 1979] is a technique commonly used in off-line cross-tabulations published by census offices. Instead of true values, rounded values are returned to the user. Introducing randomization into the rounding process is expected to enhance security. However, since correct answers can be deduced by averaging responses to queries, this randomization should in fact be a pseudo-process, producing the same responses to the same queries. Unfortunately, even with random-rounded tables, compromise is still possible. Also, rounding methods assume a static database with no user knowledge of protected values, thus their effectiveness is limited.

One interesting study on SDB security allows the system to "lie" [DeMillo et al., 1978]. Response to a query for the

median of a key-specified query set may be the value of any arbitrary record in the query set. It is shown that compromise is still possible with $O(k^2)$ queries, where k is the fixed query set size.

Protection by Random Sampling:

Sampling the database is another technique which does not always give true answers to queries [Hansen, 1971]. Only a small sample of the entire database is used for answering queries. The U.S. Census Bureau has used the principle of random sampling of records with the sampling ratio 0.001. Since the set of records is no longer selected by users, the chances of compromise is small.

Denning [Denning, 1979] proposed random sampling for on-line databases in which large samples of query sets are used for answers. Queries for frequencies and averages are computed using random samples drawn from the query sets. It is shown that the relative error in the statistics decreases as the query query set size increases; and the effort required to compromise increases with the query set size due to larger absolute errors. However the database is assumed static and users' supplementary knowledge is not considered.

Protection by Data Distortion:

In [Dalenius, 1977], the notion of "statistical disclosure" is proposed: a disclosure has occurred if, using information from a series of queries, users can estimate a database value y_i more closely than was possible without this information. Under this definition, however, disclosure cannot be prevented; it can only be controlled [Dalenius, 1977].

In one approach, records are stored together with a permanent "perturbation factor", and responses to SUM queries contain these perturbation factors [Beck, 1979]. The definition of compromisability used in [Beck, 1979] states that the SDB is statistically compromisable if it is possible to estimate data value y_i with y_i' such that

$$\text{St-dev}(y_i') < c|y_i - \text{Mean}(y)|$$

where St-dev is the standard deviation, c is a constant and Mean(y) is the mean value of all y_i's in the SDB. This approach assumes a static database and does not consider users' supplementary knowledge of protected data.

CHAPTER 3

PARTITIONING MODELS WITH DATA AND OUTPUT PERTURBATION

Variations of the partitioning model are investigated to remove some of the disadvantages of the model. Some simple security measures are defined by means of an undirected graph to help the DBA to assess the security of the SDB at a certain time. The partitioning models of this chapter will be utilized in the context of an SDB design in Chapters 4 and 5.

3.1 A Partitioning Model

Consider a database in which every record r_i has k attribute domain values and one protected domain value, v_i; each record belongs to some partition p_j, $1 \leq j \leq m$, which has k dimensions defined over k attribute domains. The database of customer accounts in Figure 2.1 of Section 2.2 is partitioned according to attribute domains income and profession, i.e. partitions are 2-dimensional. Each query q is modified to report over partition boundaries and the system returns the sum, S(q), and count, C(q), of records in partitions. In addition, changes in the database such as insertions, deletions and updates are also allowed. Clearly

this model is a variation of the SDB model in Section 2.1 in that a query set consists of records of a group of partitions.

The following assumptions are made about the database and the users.

(a) As discussed in Section 2.2, if a change is implemented as soon as it is requested then there is a danger of disclosure. Changes in a partition are assumed to be processed in pairs.

(b) An update in the attribute domain values of a record may cause the record to move from one partition to another. It is assumed that an update operation can be replaced by a pair of insertion and deletion operations.

(c) It is assumed that the deleted records are not normally re-inserted into the database, or if they are re-inserted, they have independent protected domain values. For example, for a statistical database of customer accounts in a bank it is assumed that when a data person closes his account and re-opens it some time later, the amounts in his new and old accounts are independent of each other.

(d) The user is assumed to know the properties of the database such as what the partitions are and how the system processes queries and changes. This assumption is in accord with the U.S. Privacy Act [Privacy Act, 1974]. It is also assumed that the user knows in which partition each record

belongs. In what follows, an undirected graph called the information graph is used to characterize the information revealed to the users and some of the properties of information graph are presented. It is shown that if each partition has even number of records and no record value is known initially then the database is secure.

3.1.1 An Information Graph

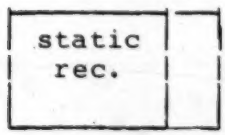
For any partition the sequence of records to be inserted and deleted forms the change sequence. Records in the change sequence are called dynamic records, otherwise they are called static records. Since record changes are processed in pairs, we form tuples of two records when they are processed together; thus the change sequence is grouped into a sequence of 2-tuples. Depending upon the operation needed for each record r_a, r_b in a tuple, there can be three different tuples, namely, (r_a^i, r_b^i) (insertion-insertion tuple), (r_a^d, r_b^d) (deletion-deletion tuple), and (r_a^i, r_b^d) (insertion-deletion tuple). Since changes in a tuple are processed at the same time, a tuple is unordered.

Assume records r_a and r_b with values v_a and v_b are both to be inserted into (i.e. the tuple (r_a^i, r_b^i)) or both to be deleted from (i.e. the tuple (r_a^d, r_b^d)) the partition p. Querying before and after the change, one can obtain the equation $v_a+v_b=c_1$ where c_1 is a constant. Similarly, the change (r_c^i, r_d^d) gives the equation $v_c-v_d=c_2$ where c_2 is a

constant. The example below further illustrates derivable equations and their equivalent form.

Example 3.1

Consider partition p_i with the change sequence (in the order of occurrence) r_1^i, r_2^i, r_3^i, r_1^d, r_4^i, r_5^i, r_6^d, r_7^i, r_7^d, r_8^i, r_8^d, r_9^d, r_3^d, r_{10}^d, r_{11}^d, r_{12}^i



partition p_i

(r_1^i, r_2^i), (r_3^i, r_1^d), (r_4^i, r_5^i), (r_6^d, r_7^i), (r_7^d, r_8^i), (r_8^d, r_9^d), (r_3^d, r_{10}^d), (r_{11}^d, r_{12}^i)

(tuples to be processed)

Derivable equations:

$v_1+v_2=c_1$ $\quad v_4+v_5=c_4$ $\quad v_6-v_7=c_5$ $\quad v_{11}-v_{12}=c_8$

$v_1-v_3=c_2$ $\quad v_7-v_8=c_6$

$v_3+v_{10}=c_3$ $\quad v_8+v_9=c_7$

where c_i's are constants.

Equivalent system of equations:

$v_1+v_2=c_1$ $\quad v_4+v_5=c_4$ $\quad v_6+v_9=c_{10}$ $\quad v_{11}-v_{12}=c_8$

$v_2+v_3=c_9$ $\quad v_7+v_9=c_{11}$

$v_3+v_{10}=c_3$ $\quad v_8+v_9=c_7$

where c_i's are constants.

Thus users may derive equations involving either sums or differences of two record values. Moreover, if there are equations $v_a-v_b=c_1$ and $v_b+v_c=c_2$ then one can replace them by the equivalent equations $v_a+v_c=c_3$ and $v_b+v_c=c_2$, where $c_3=c_1+c_2$. As long as the equations with differences have some record values in common with the sum equations, one can

repetitively change equations involving differences into equations with sums. Eventually one will arrive at an equivalent system of equations where the set of record values v_i involved in the sum equations are totally disjoint with those in the equations with differences.

In order to characterize the properties of the equivalent system of equations an undirected labeled graph is employed. An undirected labeled graph $G_i=(V,E)$ is said to be an information graph of partition p_i if V, the set of vertices, is the set of dynamic records in p_i; and E, the set of s- or d-labeled edges representing the sum and difference equations, is formed by considering each tuple in the change sequence one by one and performing the following:

(a) if the tuple to be processed is (r_a^i,r_b^i) or (r_a^d,r_b^d) then form the edge (r_a,r_b) with label s unless already formed. If the tuple to be processed is (r_a^i,r_b^d) then form the edge (r_a,r_b) with label d and

(b) repetitively replace the d-edges with s-edges, for example, if there are two edges, say (r_c,r_d) and (r_d,r_e) with labels d and s respectively, replace the edge (r_c,r_d) with the s-labeled edge (r_c,r_e).

Figure 3.1 contains the information graph G_i of partition p_i in the above example. Notice that each connected component in the graph G_i in Figure 3.1 has either s-labeled edges or d-labeled edges but not both.

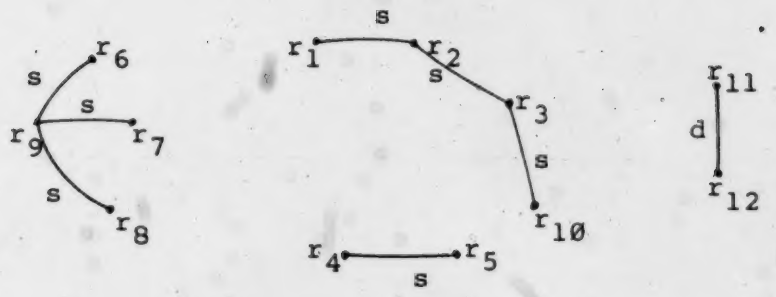


Figure 3.1 Information Graph of Example 3.1.

3.1.2 Properties of the Information Graph and a Security Result

A vertex is <u>inactive</u> if it corresponds to a deleted record in the partition, otherwise it is called <u>active</u>. Notice that because of assumption (c), a vertex can be inactivated only once. Also the information graph can be extended only from its active vertices by inactivating them. <u>End vertices</u> of a path are the two vertices at the end of a path, both with degree one. Below the properties of possible paths are specified.

<u>Property 1.</u> In an information graph, edges with label d form chains and all the vertices but one of the end vertices in such a chain are inactive.

<u>Proof.</u> Clearly s- and d-labeled edges cannot be in the same path due to the repetitive deletion of d-labeled edges performed in definition (b) of the information graph. Thus,

it suffices to show that d-labeled edges form chains with only one active vertex.

A single isolated d-labeled edge can be formed only by an insertion-deletion tuple (r_a^i, r_b^d) where r_b was in the database when the database was formed and has not been involved in any change sequence. Now the chain r_b, r_a has only one active vertex, namely r_a. It can be extended only by another insertion-deletion tuple (r_c^i, r_a^d) in which case another d-labeled edge is added to form the isolated chain r_b, r_a, r_c with r_c being the only active vertex. This procedure can be repeated only by introducing another d-labeled edge from the active end vertex. Thus only chains can be formed. #

Property 2. In an information graph, (a) there can be at most two active vertices in any path, and (b) if there exists a path with two active end vertices, the length of the path is always odd.

Proof. The proof is by induction on the number of tuples, n, in the change sequence.

Basis. The property can be easily verified for n equals 1 and 2.

Induction Step. Assume that the induction hypothesis is true for all n-tuples and that the (n+1)st tuple involves r_a and r_b.

(a) If the (n+1)st tuple is (r_a^i, r_b^i), an isolated edge

(r_a,r_b) with active vertices r_a and r_b is added to the information graph and the property is obviously true.

(b) If the (n+1)st tuple is (r_a^i,r_b^d), there are three cases.

(i) If record r_b has not been involved in any change sequence previously (i.e. existed initially), an isolated d-labeled edge (r_a,r_b) with inactive r_b is formed.

(ii) If vertex r_b is the only active end vertex in a chain u_1 with d-labeled edges (from property 1) then r_b is inactivated and r_a becomes the only active vertex in the isolated chain u_1.

(iii) If vertex r_b is connected to a path $u_2=r_m$, ..., r_j,r_b with an s-labeled edge (r_j,r_b) then two new paths with s-edges, $u_3=r_m,\ldots,r_j,r_a$ and $u_4=r_b,r_j,r_a$, are formed and r_b is inactivated. Path u_4 has only one active vertex, namely r_a. Thus the property holds for u_4. By the induction hypothesis, if there is another active vertex in u_3, it must be of odd length to r_b and hence to r_a. Also by the induction hypothesis, u_2 can have at most two active vertices one of which being r_b; thus r_b is inactivated and the new path u_3 can at most have two active vertices, one of which being r_a.

(c) If the (n+1)st tuple is (r_a^d,r_b^d), there are three cases.

(i) If neither record r_a, r_b has been involved in any change sequence then an isolated s-labeled edge (r_a,r_b) is formed.

(ii) If only one of the records, say r_a, has not been involved in any change sequence then the paths r_b belongs lose their active vertex.

(iii) If both records r_a and r_b have involved in the change sequence previously then either they are the two active vertices of an odd-length path u or they are the active vertices of two disconnected paths u_1 and u_2. For the former u will become an even-length closed path with all inactive vertices. For the latter, (1) if u_1 and u_2 are d-labeled chains then from property 1 the resultant connected chain has no active vertices; (2) if one of u_1 or u_2 initially has two active vertices and the other has a single active vertex then the new path formed by joining u_1 and u_2 will have a single active vertex; (3) if u_1 has another active vertex, say r_i, and u_2 has another active vertex, say r_j, then by induction hypothesis, r_i is at an odd length to r_a and r_j is at an odd length to r_b, thus the new path $r_i,\ldots,r_a,r_b,\ldots,r_j$ is of odd length and has exactly two active vertices, r_i and r_j. Thus the property 2 holds for this case.

Since (a), (b) and (c) cover all possibilities, the property holds for the (n+1)st tuple. #

Property 3. Cycles in an information graph are even-length and contain only inactive vertices and s-labeled edges.

Proof. From property 1 all d-labeled chains contain only one active vertex and cannot form a cycle. Thus cycles can only

be formed from s-edges by inactivating two active vertices in a path. Assume r_a and r_b are two active vertices and u is the odd-length path of s-edges containing them (from property 2). Now when the records corresponding to vertices r_a and r_b are deleted together u becomes an even-lentgh cycle containing only inactive vertices and s-labeled edges.

#

Below it is stated that if the users' supplementary knowledge does not include any protected domain value v_i, and if each partition initially has even number of records, then the partitioning model is secure.

Theorem. If no protected domain value is known initially, the partitioned database described in Table 3.1 is secure.

Proof. Information graph G_i of partition p_i contains all the equations about the dynamic records of p_i that are revealed to the user. From property 3, it is known that all the cycles in the information graph representing the system of equations are even. Such a graph is well-known to be 2-colorable [Harary, 1969]. Therefore, one can construct an infinite number of solutions for this system of equations by adding an arbitrary number to all the records which have the same color and subtracting the same amount from all the records with other color. Thus protected domain values of dynamic records cannot be disclosed. Consider the static records of partition p_i; since there is always an even

number of records in p_i (zero or ≥2), protected domain values of static records cannot be disclosed and the database is secure. #

Table 3.1 The Partitioning Model

(1) The database is divided into disjoint partitions and each partition either contains an even number of records or is empty.

(2) Changes in the database are processed in pairs.

(3) Each query is modified to report over partition boundaries and the system returns the total number of records in the partitions covered by the modified query q and the sum of their protected domain values.

Note that this security result even holds with some supplementary knowledge of users. For example, the user may know the protected domain values of all static records in p_i except two records r_a and r_b, then all the protected domain values of records in p_i are still secure in the sense that the user cannot extend his supplementary knowledge. If the users' supplementary knowledge includes the protected domain values of some dynamic and some static records in partition p_i, then all the other dynamic records which have a path to the known dynamic records in the information graph can be deduced. Moreover, if there is only one static record r_c whose protected domain value is unknown among the static records in partition p_i and if, during the process of the change sequence, partition p_i contains only r_c and the set of disclosed records, then the protected domain value of r_c may be deduced.

3.2 Partitioning Models with Data and Output Perturbation

In Section 3.1, an SDB model is presented with the following assumptions.

(1) There are even number of records in each partition.

(2) Changes in partition p_i must wait for some time until the next change in p_i.

(3) Users do not have any supplementary knowledge of protected property values.

When the above assumptions are relaxed, the model becomes insecure. Assumption (1) introduces implementation difficulties and variable-size partitions. Assumption (2) introduces an error which is dependent on the protected domain values of dynamic records waiting to be processed. Finally, assumption (3) may not always hold and may lead to a compromise. In this section, several variations to the partitioning model, which utilize data and/or output perturbation are proposed and their effectiveness in preventing compromise is analyzed.

3.2.1 A Partitioning Model with Data Perturbation

Assumptions (1) and (2) can be relaxed by introducing dummy records for each nonempty partition where the value x_j of a dummy record dr_j is a random variable with zero mean and a small variance. Thus an answer to a query may contain an error due to dummy records, but this error is pre-controlled by adjusting the mean and variance of the random

variable whereas in the model in Section 3.1, the error is dependent on the protected domain values of dynamic records waiting to be processed and is uncontrollable. Dummy records may be implemented as follows: if initially partition p_i contains odd number of records, a dummy record is inserted into the partition making the number of records in p_i even. Assume the following sequence of changes is to be made in partition p_j which initially has even number of records: r_a^i, r_b^i, r_c^d, r_d^i, ... This sequence can be changed by adding and deleting dummy records as follows: (r_a^i, dr_1^i), (r_b^i, dr_1^d), (r_c^d, dr_2^i), (r_d^i, dr_2^d),..., where dr_j, $j=1,2,...,$ are dummy records.

The distribution of the random variable x_j of dummy record dr_j must have certain properties. Its mean should be zero and different partitions should have independent random variables so that $E(\sum_{j=1}^{n} x_j)=\emptyset$ (E is the expected value and x_j, $j=1,2,...,n$, are identically distributed, independent random variables of dummy records dr_j). This property is needed since for a query involving several partitions the expected value of the error introduced due to dummy records should be zero. A normal distribution with zero mean may be a good choice for x_j.

The standard deviation of x_j is particularly important and dependent on the factors like the properties of the database, the required level of security, the accuracy of the statistical information to be revealed to the users, and

other requirements of the database system. It is the database administrator's responsibility to decide about the most suitable value for the standard deviation. For example, it should be a function of (1) sum of the protected values, v_i, in p_j (e.g. upper bounded by 0.05 of Σv_i, v_i is in p_j), (2) the distribution of v_i's in the partition (e.g. the standard deviation of x_j=the standard deviation of v_i), and (3) the protected domain value of the dynamic record with which it forms a tuple in the change sequence (e.g. assume r_a and dr_j are to form a tuple, if v_a(=annual income)=$200,000 then the standard deviation of x_j=$50,000, or if v_a(=annual income)=$10,000 then the standard deviation of x_j=$3,000).

3.2.2 A Partitioning Model with Rounding

Consider the partitioning model in Section 3.1. Assume SUM query responses for each partition are rounded using a rounding base b and let m be the true answer to the query q. Then the rounded response, S(q), is defined as

$$S(q)=\begin{cases} m & \text{if } r=0 \\ m-r & \text{if } r < \lfloor (b-1)/2 \rfloor \\ m+b-r & \text{if } r \geq \lfloor (b-1)/2 \rfloor \end{cases}$$

where $r=m - \lfloor m/b \rfloor * b$ and b is odd. Clearly,

$$m \in [S(q)-(b-1)/2,\ S(q)+(b-1)/2].$$

Now consider partition p with response S to SUM

queries. Assume S' becomes the response to the user after records r_1 and r_2 with property values v_1 and v_2 are added to p. Clearly one has

$$v_1+v_2 \in [(S'-S)-(b-1),\ (S'-S)+(b-1)]$$
$$= [kb-(b-1), kb+(b-1)]$$

Later on, if r_1 and r_2 are deleted together from p one may have $v_1+v_2 \in [k'b-(b-1), k'b+(b-1)]$

If $k \neq k'$, one can deduce a range for v_1+v_2 of size (b-2). Similarly, if a deletion and an insertion are processed together, one can obtain a range of size 2(b-1) for v_2-v_1 or v_1-v_2.

Let us now investigate the possibility of a compromise if a user knows only one protected property value, say v_1 of record r_1, in the partition p. Define the graph G'=(E,V) of p where the vertex set V is the set of records in the change sequence of p, and (r_i, r_j) is an s (d) -labeled edge in E iff r_i and r_j form insertion-insertion or deletion-deletion (insertion-deletion or deletion-insertion) tuple in the change sequence. It is easy to see that r_1 in a cycle in G' is a necessary condition for compromising the SDB as shown in the example below.

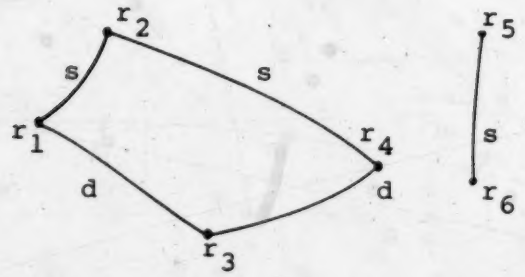
Example 3.2

Let q denote the query for p only. v_1=6 is known, b=5 and originally S(q)=13.

Change sequence: $r_1^i, r_2^i, r_1^d, r_3^i, r_5^i, r_6^i, r_3^d, r_4^i, r_2^d, r_4^d$

$(v_1, v_2, v_3, v_4, v_5, v_6) = (6,8,7,8,2,2)$

G':



Processing the change sequence leads to the range inferences $v_1+v_2 \in [6,14]$ $v_3-v_1 \in [1,9]$ $v_4-v_3 \in [1,9]$ $v_2+v_4 \in [16,24]$. Knowledge of $v_1=6$ leads to $v_2 \in [0,8]$, $v_3 \in [7,15]$, $v_4 \in [8,24]$ and $v_2 \in [8,32]$ in the given order. Thus v_2 is disclosed. #

Thus from the above example one can see that a single property knowledge may lead to other disclosures. However, the probability of its occurrence is very small. Consider G' in the above example. For each edge in the cycle one can deduce the range [MIN(x),MAX(x)] for $x=v_i+v_j$ or $x=v_i-v_j$ depending on whether it is an s or d edge. In order to have disclosure in the above situation, one must have the condition that (i) a portion of consecutive edges in the cycle have MAX(x)-x=0 and (ii) other edges in the cycle have x-MIN(x)=0.

Assume initially the sum of protected property values in the partition p is S. Two records with values v_1 and v_2 are added to p. Let $x=v_1+v_2$. We would like to find a range for x as [MIN(x),MAX(x)]. Assuming $S_b=(S) \pmod b$ and $x_b=(x) \pmod b$ are equally likely anywhere in [0, b-1] (b is

the rounding base) and independent from each other, we have, for $x_b=i$, $0 \le i \le b-1$,

$$[MIN(x),MAX(x)]=\begin{cases}[(k-1)b+1,\ (k+1)b-1] & \text{with prob. } (b-i)/b \\ [kb+1,\ (k+2)b-1] & \text{with prob. } i/b\end{cases}$$

Thus for $x_b=i$, $0 \le i \le b-1$,

$$MAX(x)-x=\begin{cases}(b-1)-i & \text{with prob. } (b-i)/b \\ 2b-1-i & \text{with prob. } i/b\end{cases}$$

giving

$$E(MAX(x)-x)=b-1$$

Similarly,

$$E(x-MIN(x))=b-1$$

Also $Prob(MAX(x)-x=0)=Prob(x-MIN(x)=0)=1/b^2$. For $b=10$, this probability is 0.01 and notice that any cycle in G' consists of at least 4 edges. Thus probability of compromise is $<< 1/b^2$ (with the assumptions of equally likely and independent x and S values).

Disclosure due to a single property value knowledge can be prevented with a simple strategy using the notion of active vertex in Section 3.1. For each path with two active vertices in G', keep adding MAX(x)-x values into y_1 and x-MIN(x) values into y_2. Clearly, any single property value knowledge of an intruder may decrease the ranges of other property values, v_i, in the path to at most y_1+y_2 and does

so only when that path turns into a cycle. Thus the database system may prevent the formation of a cycle using dummy records if $y_1+y_2=\emptyset$.

3.2.3 A Partitioning Model with Rounding and Data Perturbation

Consider the partitioning model in Section 3.2.1, i.e., insertions and deletions are processed together with dummy records. SUM query response, S(q), which covers partitions p_i, $1 \le i \le j$, is defined as follows. $S(q) = \sum_{i=1}^{j} S'_i$, where S'_i is the rounded sum of property values of records and the dummy record in p_i.

Clearly this model removes assumptions (1) and (2) and is better than the partitioning models in the previous two sections (3.2.1 and 3.2.2). Below it is shown that under normal circumstances, the expected error in this model is zero.

Assuming N records equally likely anywhere in the database and n partitions, the probability that the number of records, Z, in partition p is k is given by

$$\text{Prob}(Z=k)=p_Z(k)=\frac{\binom{N}{k}(n-1)^{N-k}}{n^N} \qquad k=\emptyset,1,\ldots,N$$

where p_Z is the probability function for Z.

Thus Z has a binomial distribution, and its mean and variance are

$$M_Z = \frac{1}{n^N} \sum_{k=1}^{N} \binom{N}{k} k(n-1)^{N-k} = (N/n) \qquad (1)$$

$$Var(Z) = N(1/n)(1-1/n)$$

Let $V_1, V_2, \ldots, V_k$ be a sequence of independent and identically distributed random variables (r.v.) representing the property values of records in p, and the mean and variance of these r.v.s be denoted by M_V and Var(V). Let T be a r.v. for the dummy record in p with mean $M_T=\emptyset$ and variance Var(T). The sum of property values of records in p is defined as $S=\sum_{i=1}^{Z} V_i+T$. Assuming Z is independent of V_i's and T, we have

$$E(S) = E(E(S|Z))$$

and $$E(S|Z=k) = kE(V) + M_T = kM_V$$

giving $$E(S|Z) = ZM_V$$

and $$E(S) = E(ZM_V) = M_Z.M_V \qquad (2)$$

Also $$Var(S) = E(Var(S|Z)) + Var(E(S|Z))$$

but $$Var(S|Z=k) = kVar(V) + Var(T)$$

$$Var(S|Z) = zVar(V) + Var(T)$$

Hence $$Var(S) = E(ZVar(V) + Var(T)) + Var(ZM_V)$$

$$= Var(T) + M_Z.Var(V) + Var(Z).M_V^2$$

$$= Var(T) + (N/n).Var(V) + N(1/n)(1-1/n).M_V^2 \qquad (3)$$

Let $R=(S)_{mod\ b}$. Thus the response to a SUM query for partition p will be (S+b-R) if R > (b-1)/2 or (S-R) if R ≤ (b-1)/2. Then the error W will be

$$W = \begin{cases} T + b - R & \text{if } R > (b-1)/2 \\ T - R & \text{otherwise} \end{cases}$$

If V_i's and T are symmetrically distributed about M_V and M_T respectively, we have

Lemma. If $M_V.M_Z = kb$, k ∈ Integers, b is odd, then

(a) $M_W = \emptyset$

and (b) $Var(W) = Var(T)$

Proof. See Appendix A. #

Thus if the database system has control over the size of the partitions (which may not be possible) and V_i's are symmetrically distributed about M_V, M_Z (given by (1)) can be adjusted to obtain $M_W=\emptyset$, i.e., the expected error introduced in a query is zero.

Notice that for a partition with relatively large number of records, using the central limit theorem [Mood et al., 1974], S is normally distributed (and hence symmetrical about M_S) even if V_i's are not symmetrically distributed. Thus the assumption of symmetrically distributed V_i's are

not needed for partitions with sufficiently large number of records. Moreover, since the variance of error is equal to Var(T), (i) it can be adjusted along the line of suggestions in Section 3.2.1, and (ii) Var(T) may be increased for paths with y_1, y_2 = Ø (described in Section 3.2.2) in order to have better protection.

3.3 Security Measures Related to the Information Graph

The information graph in Section 3.1 has another usage. Clearly, users' knowledge of one property value of an individual r in the model in Section 3.1 is sufficient to disclose all other property values of individuals that are in the same connected component with r in the related information graph. Thus, a measure of security may be defined in terms of the number of connected components of information graph. The reachability set Rs is defined as a subset of the individuals in partition p such that they are in the same connected component. We also define

$$\text{Reachability Constant } w_1 = \frac{\text{no. of Rs in the inf. graph}}{\text{no. of vertices in the inf. graph}}$$

Clearly $\emptyset < w_1 \leq 1$, and $w_1 \cong 1$ implies relatively "more" security (more connected components in the information graph) and $w_1 \cong \emptyset$ implies relatively "less" security. Another security measure may be largest reachable set size w_2, i.e. $w_2 = MAX|Rs|$ for all reachability sets Rs. Thus relatively small w_2 and $w_1 \cong 1$ implies relatively "more"

security. For example, if the user's knowledge includes x dynamic records of partition p, he can at most increase his knowledge by $(w_2-1)x$ more protected domain values.

The two described measures of security, w_1 and w_2, are not controllable in the model described in Section 3.1. For some databases, dummy records may be used to control and change the sizes of the reachability sets of records and, thus, control the security measures w_1 and w_2. In Appendix B, some ways of applying dummy records to control those paths with too many vertices are briefly discussed.

CHAPTER 4

STATISTICAL DATABASE DESIGN

The SDB Security problem is investigated at the conceptual model level. Three different data models are analyzed for their suitability as a conceptual model of SDB. Using a formal framework, the design of an SDB is investigated. Possible types of inferences are classified, security constraints are defined, and enforced. Implementation issues of the design are discussed.

4.1 Introduction

Below some shortcomings of previous studies on SDB security are discussed.

1) Statistical databases provide statistical information about groups of individuals in the real world. The assumption is that statistical information about a group of individuals conveys a meaningful aspect of that group of individuals. However, statistical information about an arbitrarily chosen group of individuals may not have a useful meaning attached to it. Previous SDB security studies have not dealt with the question of whether or not

statistical information is "meaningful", and some have set forth questions (and given answers) with assumptions like "every possible combination of records can be requested" or "all possible medians of any sets of records are queriable" [Dobkin et al., 1977; Dobkin et al., 1979; Reiss,1979]. These types of assumptions cause an explosion in the complexity of the problem, and consequently,the protection measures highly limit the richness of the SDB (see Chapter 3). However once a proper definition of the "statistical information" is used, and an analysis of the portion of the real world represented by an SDB is made for determining its statistical information, these combinatorially explosive possibilities perhaps can be reduced or even eliminated.

2) The SDB models used in previous studies (see Chapter3) used terms like records, record fields, etc.. Databases are more than collections of records, and the information in databases may be highly complex. Databases contain a model of some portion of the real world; effectiveness of protection measures, if treated at that level, may increase. In all previous studies, the SDB models used were closer to the physical level, rather than the conceptual level, of the database. Thus they encountered the problem of security at a very low level, that of physical records. Although these studies have contributed to our understanding of the problem, their SDB models were incomplete, and the results were usually negative in tone.

3) All previous studies (except [Yu and Chin, 1977]) considered static databases in order to simplify the problem. Changes may occur not only at the level of insertions, deletions or updates of individuals (i.e., primitive changes as discussed in Chapter 3), but at the level of the conceptual model (i.e., high-level changes such as different views, abstractions, etc.). The problem of SDB security should also be investigated for dynamic databases to capture the dynamics of the real world.

4) In the real world, users' additional (supplementary) knowledge may take the form of general rules, relationships, or simply, protected property values. If the database administrator (DBA) is aware of this information, effective security measures can be imposed easily. The example below illustrates this situation.

Example 4.1 Consider a database of employees of a certain computer manufacturing company in which the sum of salaries of employees is queriable. Assume the following information (which is not represented in the database and hence unknown to the database system) exists.

(a) Salary range of a new systems analyst with B.S. is $[10K,12K].

(b) Salary range of a new systems analyst with M.S. is $[12K,14K].

Now assume two new systems analysts are hired and

information about them is inserted into the database. If the change in the sum of salaries of systems analysts is $27K, then users can conclude that the new employees have Ms degrees.

Most problems in SDB security can be removed by a good model of the real world environment so that the DBA can take effective protection measures. Thus existing relationships and semantics of the information should always be considered for an effective SDB design.

5) When users' additional knowledge increases, some mechanisms are are needed for the DBA to decide (i) what other information has been disclosed by users, and (ii) what protective measures should be taken. In other words, exact information revealed to users should be kept in some compact form for auditing purposes. Some previous studies proposed investigation of log trails for auditing [Hoffman and Miller, 1970; Dobkin et al., 1977; Hoffman, 1977]. However, for very large databases, the enormous amount of information in log trails is of little help for checking security (not to mention the "masking" of queries by users [Denning et al., 1979; Schlorer, 1976]).

4.2 Statistical Database Design

This section gives formal definitions of the SDB and the security problem (section 4.2.2) and discusses the design of an SDB which employs security constraints at the

conceptual data model level. Below are the desirable features of the SDB design in terms of the "goodness" criteria introduced in Section 2.2.

(a) Effectiveness of the protection. In order for the SDB system to be effective, the database should be equipped with the following information.

1) A "good" conceptual model. As a response to problem 2 in Section 4.1, the SDB security should be elevated to the conceptual model level.

2) Well-defined statistical information. Statistical information must be well-defined and an analysis of the specific information and its statistical constituents should be made. This will help to reduce the size of the security problem (crystallize the complex relationships, define the information to be secured, etc.) and thus eliminate problem 1 mentioned before.

For the real world model, the statistical information revealed to users will only be about pre-defined groups of individuals. The intersection of these groups of individuals will give a set of indivisible groups of individuals and any statistical information about these indivisible groups of individuals will constitute atomic information. Thus the database system is no longer interested in giving out uncontrolled, random statistical information to users, which may easily be exploited, but rather it will give out well-defined information that can at most be reduced to atomic information.

3) Controlled changes in the database. The DBA should be equipped with data manipulation operators, and dynamics, as well as statics, of the real world should be revealed to users (problem 3). However this should be done in a controlled manner and the information revealed due to the changes in the environment should be recorded for auditing. (Note that 1974 US Privacy Act [Privacy Act, 1974] necessitates the inclusion of changing aspects of the environment.)

4) Information about users' additional knowledge. Users' additional knowledge should be maintained and kept up-to-date in the SDB. It is assumed that the DBA is correctly informed about users' additional knowledge of protected information.

b) Efficiency of the protection. Below features of the SDB to improve the efficiency of the protection are described.

1) Disjoint user groups should be defined to utilize the fact that their initial knowledge may be substantially different from each other or they may not necessarily have the same access authorization to different parts of the database.

2) Different levels of statistical information should be revealed to different users. For example, some users may not be allowed to access certain detailed statistical information.

3) For each group of individuals about which

statistical information is to be revealed, allowable statistical query types are defined. This leads to different security constructs and mechanisms for different types of statistical information.

c) Richness of the information revealed to users. Clearly, investigating the security problem at the conceptual model level provides the database designers with more control over the richness and usefulness of the SDB. However, atomic information should not be further decomposable by templates or by queries such as join, select and project operators in a relational model [Codd, 1970; Codd, 1974]. It is also assumed that the DBA confirms the security and compatibility of any new view before granting access to it.

4.2.1 Constituents of Statistical Information

Statistics studies specific aspects of individuals in a population which may be conceptual or physical. The individuals in the population have something in common so that they altogether form the population. Most statistical methods can be viewed as ways of making inferences about a population. Such inferences are made after the examination of a "sample" from the population. A database may contain the whole population or a sample of the population. The user may or may not use the statistical information for statistical inferences. In any case the central concept is the population concept. For the specific environment at

hand, once the populations to be studied are identified then the individuals are no longer important, and two individuals with nothing in common will not be included in the answer of the same statistical query.

The database system should also differentiate the quantitative properties of individuals for which statistical information is to be revealed and the defining characteristics of a population. For example, "sum of salaries of employees" is a quantity related with the employee population but "sum of salaries of employees where salary >$12K" gives information about a different population. Not distinguishing this difference may cause protection problems. Similarly, for example, "number of employees" and "number of employees convicted of felony" give information about two different populations.

4.2.2 Formal Definitions of the SDB and the Security Problem

In order to provide insight to the features of the SDB and a framework within which to investigate the SDB design, formal definitions of the SDB and the security problem are given in this section.

In general, there may be two different approaches in modeling a database system: set theoretic models or finite state models. For finite state modeling there may be two approaches: (a) state snapshot, in which the rules are given to define valid states of the database, (b) state

transition, in which legal database operations are given, and these operations are guaranteed to preserve the security of the SDB. In this section, a finite state model of an SDB with state transition approach is described.

The database system models some portion of the real world which is called the application. The application can be thought of as having a state and certain allowed transitions between states. The application state represents a "snapshot" of the application at a given time. An application is represented in the database system by an SDB data model.

SDB Data Model:

SDB Data Model is a system of 4-tuples

(Schema, Query Types Set, Query Mapping Function, Operation Types Set)

Schema contains descriptions of populations and their properties. For example, schema for the D-A Model [Smith and Smith, 1977a] contains hierarchies of object types. The Schema for the E-R Model [Chen,1976] contains the descriptions of entity sets and relationship sets. The Query Types Set contains allowable statistical query types such as MAX, MIN, SUM, MEDIAN, etc.. The Query Mapping Function identifies which statistical query types are allowed for properties of populations in the schema, i.e.,

Query Mapping Function : Properties of Populations --> Subsets of the Query Types Set

Operation Types Set contains two different groups of operation types:

Operation Type-1 i : Schema x Arguments --> Schema $1 \leq i \leq m1$

Operation Type-2 i : Schema x Arguments x Database State ---> Database State $1 \leq i \leq m2$

Type-1 operations correspond to high-level schema modification operations such as "decompose population" or "delete population", etc.. Type-2 operations correspond to low level operations such as insertion, deletion or update of individuals. Notice that (unlike in [Borkin, 1978]) the schema is not assumed to be static. The Database State of the SDB consists of each user group's knowledge set (to be described) and a representation of the application state. The representation of the application state may be a mapping from the schema to the subsets of individual objects and relationship tuples specifying the individuals belonging to the populations in the schema.

Given a schema and a set of arguments for operation type-1, one can generate operation (i) corresponding to argument (i) as

Operation (i) : schema --> schema

Or given a schema, a set of arguments and the database state, one can generate operation (i) corresponding to argument (i) as

Operation (i) : Database State --> Database State

SDB Knowledge Base:

SDB Knowledge Base is a system

(User Groups Set, Knowledge Sets,
Knowledge Base Operation Types Set)

User Groups Set consists of user groups, and each user belongs to one user group. For each user group, there is one Knowledge Set containing the users' additional knowledge about the application, which is in the form of explicit facts and general rules.+ Explicit facts can be represented by a set of predicates describing either the relationships between entities in the application or the property values of entities. Knowledge Base Operation Types Set contains available knowledge base operation types, and given a set of arguments and a knowledge set, one can generate the operation corresponding to any operation type as

Operation : Knowledge Set --> Knowledge Set

+An example to general rules may be "Every programmer has a B.S. degree". [Minker, 1978] refers to general rules as "axioms".

Since the only database states of concern are those which can be reached by the set of allowed operations, the set of valid database states can be defined as consisting of some initial state and those states consisting of the closure of the SDB data model's set of allowable operations and knowledge base operations of user groups.

Statistical Database:

The SDB specifies the SDB data model, the current database state, all possible state transitions and a set of security constraints about the representation of the application state. Security constraints are dynamic conditions that are always satisfied at any database state. Security constraints are dependent on users' additional knowledge, representation of the application state, the query mapping function, etc.. Security constraints, when applied to the user group's statistical queries, are in the form of either "suppress user group u's statistical queries of type i for population p if condition C holds" or "remove individuals x,y,...,z from statistical queries of type i for population p if condition C holds". Thus the SDB is a system

(Data Model, Database State, Security Constraints)

Security Problem of the SDB:

For each user group, the set of explicit facts relevant to the application represented by the SDB is classified as

confidential (F_c) or nonconfidential (F_n) and known (F_k) or unknown (F_u)(See Figure 4.1). (Note that F_k is represented by the Knowledge Set and the representation of the application state.) Compromise (or disclosure) occurs if a user in a user group changes $F_{u,c}$ (i.e. F_u & F_c) into $F'_{u,c}$ where $F'_{u,c} \subset F_{u,c}$[+]. Similarly, the set of general rules is classified as known (I_k) and unknown (I_u), and the unknown set of general rules is classified as strongly-compromising (I_{st}), weakly-compromising (I_w) and safe (I_{sf}) as follows:

(1) Let p_i, $1 \leq i \leq k$, and $k \in$ integers, be a disclosure procedure which, in order to compromise the SDB, uses only some known facts and general rules and a nonempty subset I_i of the unknown general rules set, I_u. Then the strongly-compromising set of general rules is defined as $I_{st} = \bigcup_{i=1}^{k} I_i$

(2) Let p_j, $1 \leq j \leq t$ and $t \in$ Integers, be a disclosure procedure which uses only some known facts and general rules, a nonempty subset, $F^j_{u,n}$, of unknown nonconfidential facts and a nonempty subset I_j of unknown general rules. Then the weakly-compromising set of general rules is defined as $I_w = \bigcup_{j=1}^{t} I_j - I_{st}$. Moreover we define $F'_{u,n} = \bigcup_{j=1}^{t} F^j_{u,n}$. Clearly, $F'_{u,n} \subseteq F_{u,n}$.

[+] $\subset$ means "proper subset".

The set of safe general rules is defined as

$$I_{sf} = I_u - (I_w \cup I_{st})$$

Clearly, at any valid database state, F_k is closed over I_k in the sense that any disclosure procedure using only known facts and known general rules can only infer known explicit facts.

For the security of the SDB, the set of unknown confidential facts, $F_{u,c}$, and the strongly-compromising set of general rules, I_{st}, should be protected. Moreover the database system should protect either $F'_{u,n}$ or I_w or a subset $F''_{u,n}$ of $F'_{u,n}$ and a subset I'_w of I_w such that there does not exist any disclosure procedure which uses only $(F'_{u,n} - F''_{u,n})$, F_k and $(I_w - I'_w)$.

The security problem of SDB can now be defined as follows: the SDB is secure at a database state iff F_k is closed and the database system ensures protections of $F_{u,c}$, I_{st} and either $F'_{u,n}$ or I_w or ($F''_{u,n}$ and I'_w).

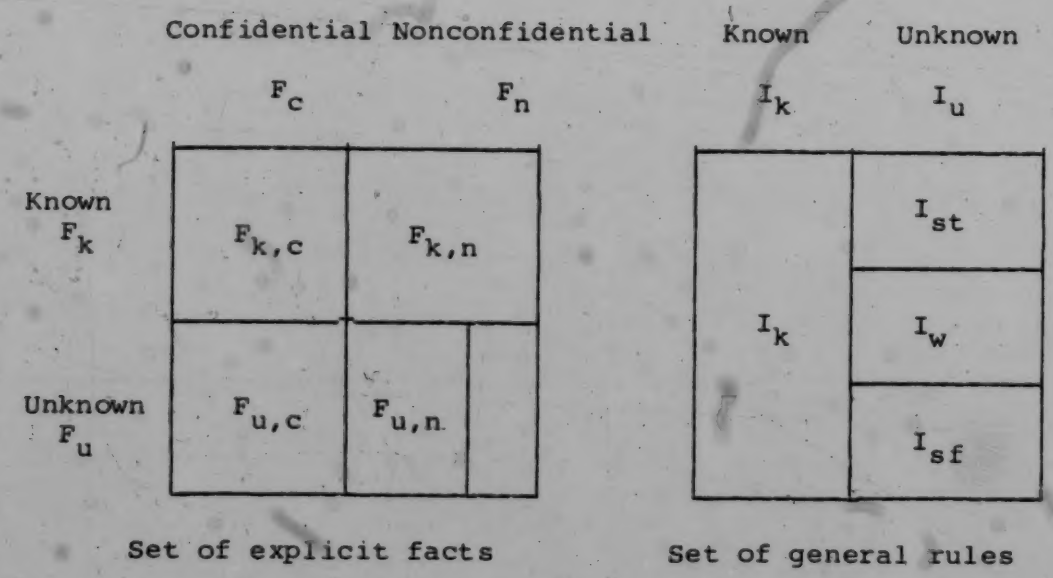


Figure 4.1. Classification of explicit facts and general rules relevant to the application

4.2.3 Conceptual Data Models for SDB Design

The SDB data model should be similar to the conceptual data model of any other general-purpose database. There are several reasons for this requirement, besides effectiveness of the protection and the richness of the SDB.

1) For some users and at least for the DBA, the SDB is just a normal database and these users should have access to all information in the database (not just aggregates).

2) It is necessary to have total information about the environment in order to enforce security, integrity and validity in the database.

Although Section 4.2.2 gives the definition of the SDB data model it does not advocate any specific data model. With the recently renewed interest in conceptual data models, over thirty different data models are mentioned in [Kerschberg et al., 1976; Nijssen, 1976]. From the security viewpoint, our concern is twofold. We are concerned about the structure of the conceptual data model in order to define atomic information and to give out controlled statistical information. We are also concerned about the semantics and the redundancy of the conceptual model in order to successfully mirror the real world environment so that (a) the security measures can easily and naturally be provided and (b) the database is still a highly rich and useful one for users. Thus a structured, semantic and redundant conceptual model for the SDB is required. In this

chapter, the Data Abstraction Model (D-A Model) [Smith and Smith, 1977a; Smith and Smith, 1977b] will be used in the design of the SDB. The choice of the D-A Model among other structured, semantic and redundant data models is motivated by the ease in applying protection measures without bringing many extra constructs and restrictions to the conceptual model. However, the SDB design may easily be modified for any other structured, redundant and semantic data model; and below, two other data models, namely, the Entity-Relationship Model [Chen, 1976] and the Extended Relational Model [Codd, 1979], are discussed for their suitability as a conceptual model of the SDB. It should be noted that the aim is to investigate the needed modifications (i.e. rules and constructs) in order to define populations clearly and to analyze and control inferences. Thus the data models are only briefly described, and other issues such as expressive power, semantics, naturalness, etc., of the data models are not discussed.

The Data Abstraction Model:

In this section, the D-A Model for SDB design is summarized and modified. Our goals are to augment the conceptual data model with the population concept and to identify atomic information.

Smith and Smith introduce two kinds of database abstractions. Aggregation (naming relationships) is an

abstraction which turns a relationship between objects into an aggregate object. Generalization (naming classes) is an abstraction which turns a class of objects into a generic object. All objects (individual, aggregate, generic) are given uniform treatment in the D-A Model. The real world is modeled as a set of aggregation hierarchies intersecting with a set of generalization hierarchies. Abstract objects (i.e. generic and aggregate objects) occur only at the points of intersection. In the context of the relational model [Codd, 1970; Codd, 1974], the D-A Model is proposed as a conceptual model. Our aim is to modify the generalization hierarchy in such a way that all populations are identified in a systematic manner and a generic object in the hierarchy consists of a (group of) population(s).

In the D-A Model, for a class of individual objects corresponding to a generic or aggregate object G, the set of attributes (or properties) which are common to all individual objects are called G-attributes (or G-properties). Clearly, individual objects of all generic objects that are descendants of G in the generalization hierarchy also have the same attributes.

Consider the same example of employees of a certain computer manufacturing company. Figure 4.2 illustrates one particular decomposition of Computer Scientist into lower level generic objects. Notice that there are two mutually exclusive groups of partitions (also called clusters) of

Computer Scientist, one is {Programmer, Systems Programmer, Systems Analyst} and the other is according to the degree obtained. Now assume that one also has the "country in which Ph.D. was obtained" information about Computer Scientists. Clearly one may ask about the "population of US-educated systems analysts with Ph.D.". In [Smith and Smith, 1977b] this information is kept as an attribute of objects in the generalization hierarchy and there is no provision for further partitioning. The reason for this is that each abstract object is required to be explicitly named using natural-language nouns (e.g. Programmer, Systems Analyst, etc.) and these names help us to relate our understanding of the real world with its intended reflection in the relation definition. However, the generic object "US-educated systems analyst with Ph.D." is certainly described by a phrase, not by a natural language noun and yet we are interested in this particular object and it has to exist in the hierarchy. Thus for SDB design purposes we will take more freedom at this point and use phrases to describe populations. Figure 4.3 contains the partitioning of Systems Programmer with Ph.D. and Systems Analyst with Ph.D. according to the attribute "country in which Ph.D. is obtained".

Now assume that we also have the "years of programming experience" information for Programmers. Populations using this information may be formed, such as "programmers with 5 years of programming experience" or "programmers with Ms and

2 months of programming experience", etc.. At this stage a design decision problem appears. If the "years of experience in the company" information uniquely identifies many individuals by creating large numbers of populations with single individuals, the security is endangered. We assume that SDB designers make the decomposition decisions using their knowledge of users' needs, i.e. if there are very many populations each with few individuals, then the designers will cut down the number of populations and still preserve a good model of the real world. However this does not mean that initial design decisions cannot be changed, indeed, if a need arises, some mechanisms will be available to the DBA so that, with an assessment of the security of protected information, the decomposition of objects may be changed some time later. Figure 4.4 shows one particular design decision about the usage of "years of programming experience" information for decomposing the object Programmer.

In the SDB, statistical information about individuals in a population is made available to users. Clearly, each abstract object in the D-A model forms a population of individual objects. We call smallest nondecomposable group of individuals an Atomic Population (A-population). For example, in Figure 4.5, ASSIGNMENT-IN-DATABASE-PROJECT, PROGRAMMER and PROJECT are A-populations. In order to preserve the indivisibility property of A-populations the

following rule is applied: any population corresponding to any abstract object in the model is composed of mutually exclusive A-populations that explicitly exist in the model (Rule 1). The restriction that A-populations explicitly exist in the conceptual model may bring limitations to the richness of the SDB. However, it is needed to provide systematic assurance of the security of protected information in the SDB.

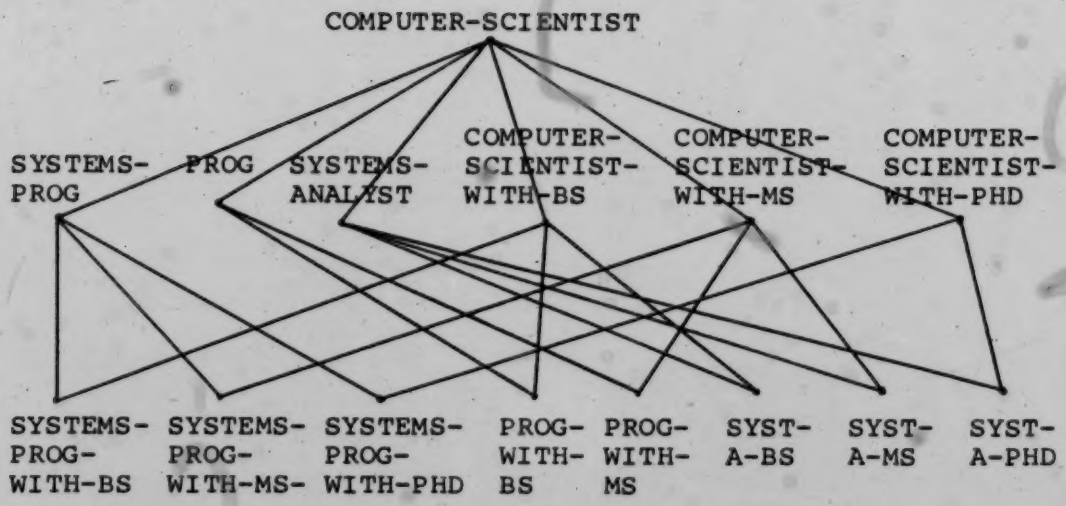


Figure 4.2. Decomposition of the generic object Computer Scientist

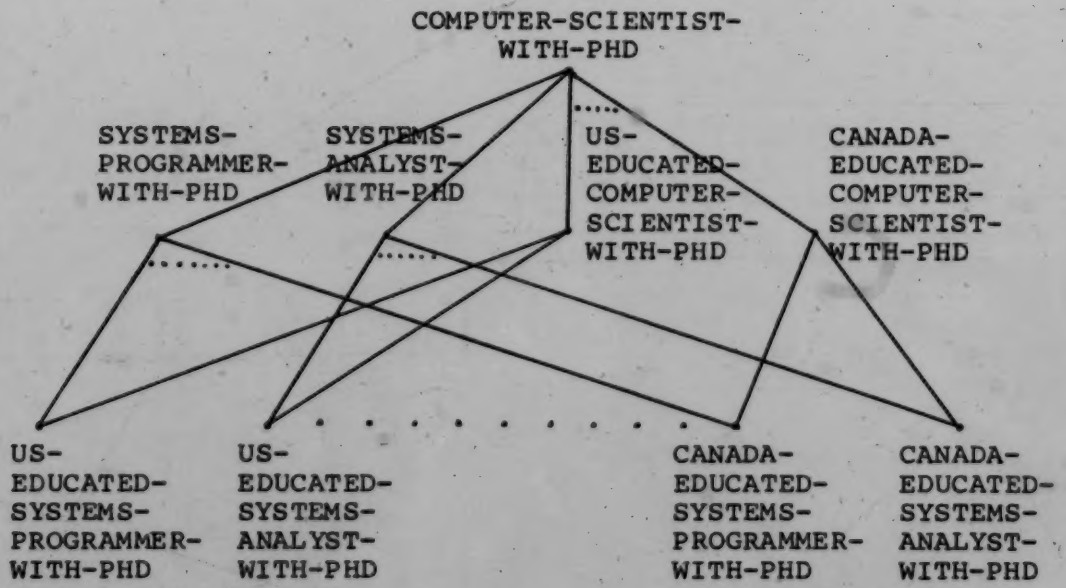


Figure 4.3. Decomposition of the generic object Computer Scientist with PhD

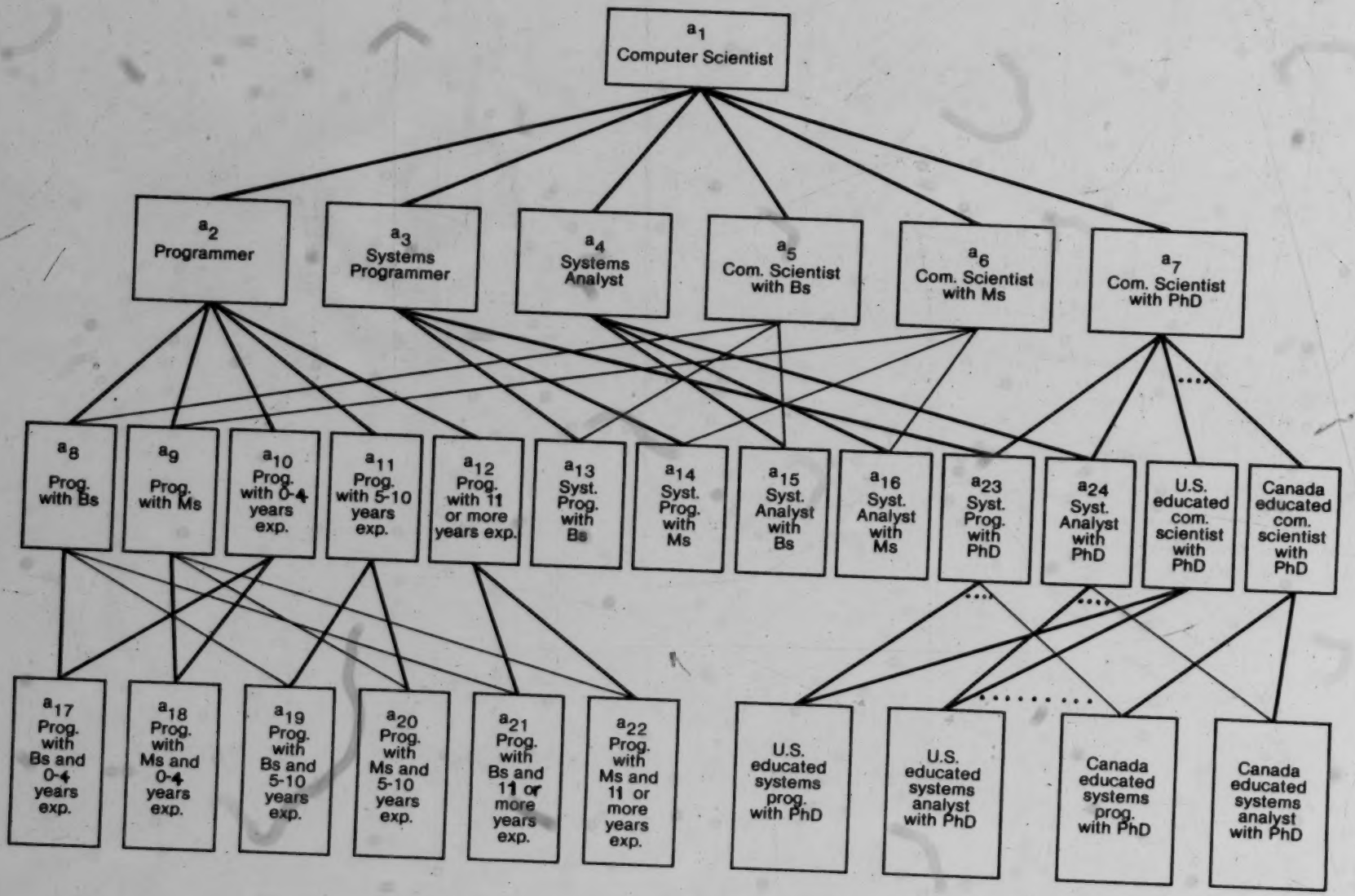


Figure 4.4 Generalization Hierarchy for Computer Scientist.

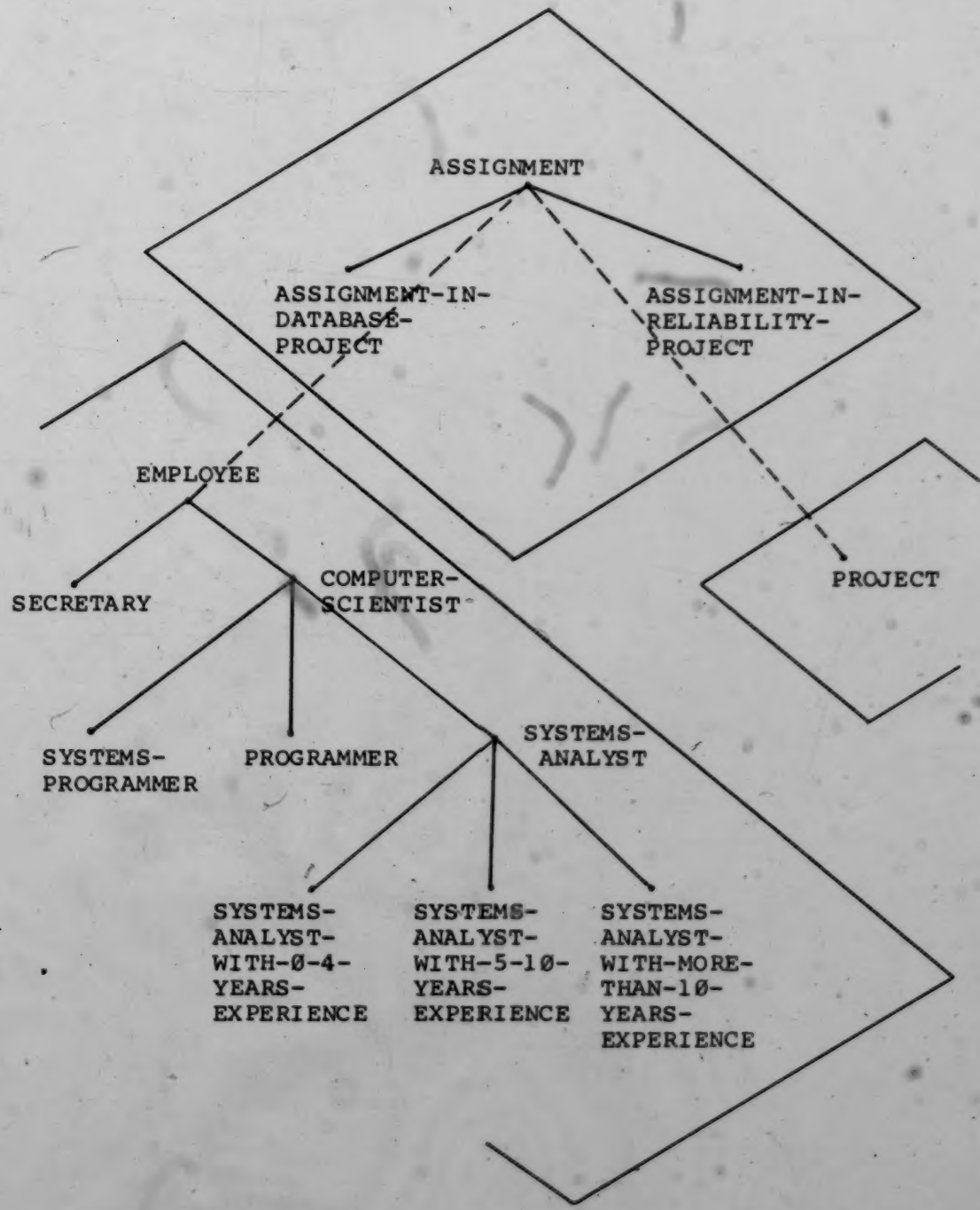


Figure 4.5 Database of employees, projects and assignments

The Entity-Relationship (E-R) Model:

The E-R Model adopts the view that the real world consists of entities and relationships. An entity is a thing which can be distinctly identified. A relationship is an association among entities. Entities are classified into different entity sets such as EMPLOYEE, PROJECT and DEPARTMENT. Similarly relationships are classified into relationship sets such as PROJECT-WORKER and DEPARTMENT-EMPLOYEE.

In the SDB, entities and relationships correspond to individuals, and each entity set or relationship set is a population. Statistical information about values in attribute-value pairs of entities or relationships are revealed to users. However, in order to define A-populations and to enforce security constraints, we need additional rules and constructs as described below.

1) Some means are needed to identify the A-populations that a population contains. For example, the fact that entity set MALE-PERSON is a subset of the entity set PERSON should be easily accessible to the database system. This is needed, for example, when constraints applied to individuals in an A-population are also applied to all populations that include the same A-population (the details of this requirement will be discussed in the later sections).

2) Each entity set or relationship set must be composed

of some mutually exclusive entity-sets or relationship sets (rule 1).

3) The protection mechanism of the SDB should be able to locate all populations that contain a given A-population. This is needed, for example, while processing insertions, deletions or updates of individuals. In the E-R Model, locating all populations containing a given A-population requires additional structures or rules.

Finally, in the E-R Model, the job of defining the allowable types of statistical queries in a systematic manner relies on the DBA whereas this task is easier in the D-A model because of its hierarchical structure (see Section 4.2.4).

The Extended Relational Model (RM/T):

In RM/T, there are entities and entity types classified by whether they

1) fill a subordinate role in describing entities of some other type, in which case they are called characteristic,

2) fill a superordinate role in inter-relating entities of other types, in which case they are called associative,

3) neither of the above, in which case they are called kernel.

Using these entity types, the semantic structures defined

are characteristic tree, association graph, cartesian aggregation, generalization and cover aggregation. Cartesian aggregation is the aggregation abstraction of the D-A Model. Below each of the new semantic structures of RM/T and related SDB issues as to how to obtain populations and indivisible A-populations are discussed.

a) Characteristic tree. The characteristic entity types that provide description of a given kernel entity type form a characteristic tree. Example 4.2 below is from [Codd, 1979].

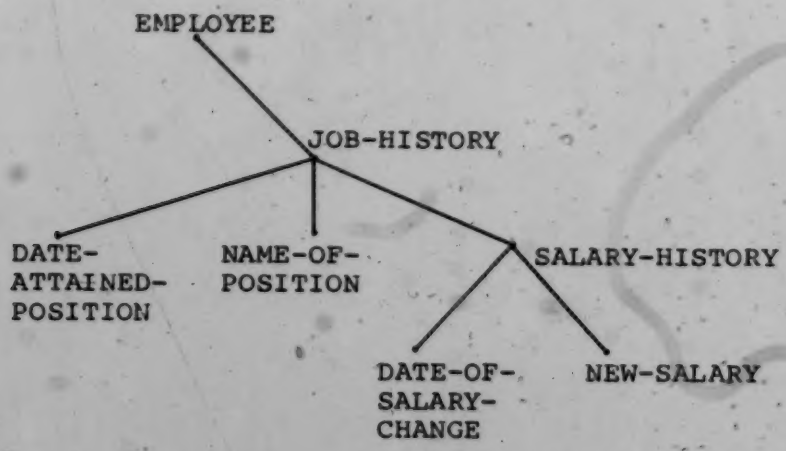


Figure 4.6. A Characteristic Tree

Example 4.2 EMPLOYEEs (a kernel entity type) have a JOB-HISTORY (characteristic entity type subordinate to EMPLOYEE) whose immediate properties are DATE-ATTAINED-POSITION and NAME-OF-POSITION (see Figure 4.6). This information is augmented by SALARY-HISTORY (characteristic entity type

subordinate to JOB-HISTORY) whose immediate properties are DATE-OF-SALARY-CHANGE and NEW-SALARY. The mapping between entities in the parent and child nodes is one-to-many, e.g. one EMPLOYEE has many JOB-HISTORY entities and one JOB-HISTORY entity has many SALARY-HISTORY entities.

Each of EMPLOYEE, JOB-HISTORY and SALARY-HISTORY nodes forms a population. A-populations of these populations may be formed by decomposing them (as generalization abstractions) using their properties, their mapping to the parent nodes, etc.. For example, SALARY-HISTORY may be decomposed using (a) EMPLOYEE individuals, (b) NAME-OF-POSITION of JOB-HISTORY, (c) date, (d) salary ranges. Notice that, for the SDB, if the decomposition of a population in the characteristic tree is effected by its parent populations, some limitations may be brought on the information revealed to users in order to prevent compromise.

b) Association Graph. An associative entity inter-relates entities of other types, and this inter-relation is represented by the association graph in the RM/T. If the association among individual entities is to be protected then some limitations should be brought on the information revealed to users about the associative entities or the entities that they inter-relate.

c) Generalization. Codd [Codd, 1979] renames the

generalization abstraction of the D-A Model as Unconditional Generalization Inclusion (UGI), and also describes an abstraction called Alternative Generalization Inclusion (AGI), which is an alternative or conditional inclusion of entities of an entity type into some other entity types. Clearly, the only structural difference between the UGI and the AGI is that the AGI decomposes a population P into mutually exclusive populations that are one level above P in the hierarchy. Thus controlling inferences for the AGI is the same as the UGI.

d) Cover Aggregation. Cover aggregation is an aggregation in which a subset of entities of the same type forms another entity with a different entity type. For example, a CONVOY-OF-SHIPS is a cover entity of entity type SHIP, or a CLUB that some people belong to forms a cover aggregate of PEOPLE.

For the SDB design, cover aggregate entity types partition the group of entities resulting in smaller populations. Intersection of these smaller populations gives A-populations of covered individuals.

4.2.4 Statistical Information Related to each Population

For each population in the conceptual model of the SDB, the following is defined. (Note that (a) and (b) below are defined by the schema and the Query Mapping Function, respectively,in the formal definition of the SDB in Section

4.2.2).

(a) the properties of the population[+] for which statistical information is to be revealed, e.g. SALARY or ABSENT-DAYS for the population EMPLOYEE,

(b) whether COUNT queries requesting the number of individuals in the population are permitted, and

(c) the allowable types of statistical information for each property of the population which may be one or more of

MEAN,

SUM,

MAX, MIN, MEDIAN, K-LARGEST (order statistics)

VARIANCE,

STANDARD DEVIATION,

k-MOMENT, k=2,3,...

Clearly, if, in Figure 4.4, SUM query for the SALARY property of PROGRAMMER and SYSTEMS-PROGRAMMER is allowed then SUM information for the SALARY of SYSTEMS-ANALYST is deducable. Thus, unless individual security needs of populations require otherwise, the following two rules are found necessary for the uniformity of the revealed statistical information and richness of the database.

Rule 2) The allowable set of statistical query types

[+]By "the property of a population", "the property of individuals in the population" is meant.

should be identical for the same property of all populations in the same cluster. (Subpopulations created by a mutually exclusive decomposition of a population in the generalization hierarchy form a cluster).

Consider Figure 4.4. Assume that SALARY is an attribute of all the objects in the hierarchy and SUM query is allowed for SALARY of Programmer. SUM query should be also allowed for SALARY of Systems Programmer and Systems Analyst.

Rule 3) The allowable set of statistical query types for a property of any population should be the subset of the allowable set of query types for the same property (if it exists) of its father population in the generalization hierarchy.

Consider Figure 4.4. Assume the statistical query SUM of SALARY is allowed in populations Programmer, Systems Programmer, Systems Analyst and statistical query MEDIAN of SALARY is allowed in populations Computer Scientist with Bs, Ms and PhD. Statistical queries SUM and MEDIAN are allowed for the population of Computer Scientist.

Since COUNT queries do not directly reveal information about protected properties of populations, applying protection measures down to A-populations may unnecessarily restrict the richness of the SDB. Thus, a <u>security atom population</u> (SA-population) is defined to be the largest population such that no statistical information about any

property of any of its proper subsets can be revealed to users. Notice that an SA-population contains one or more A-populations. The set of values to be protected for each property in a SA-population is called a security atom value set (SA-value set). The following example illustrates SA-populations.

Example 4.3 Consider Figure 4.4. Assume there are only two protected properties, SALARY and ABSENT-DAYS, and

(i) for all populations, COUNT query is allowed.

(ii) SUM query for SALARY and ABSENT-DAYS is allowed for populations a1, a2, a3 and a4. MEDIAN query for SALARY is allowed for populations a1, a5, a6 and a7. Also, SUM query for ABSENT-DAYS is allowed for populations a1Ø, a11 and a12. Clearly, populations a8, a9, a13, a14, a15, a16, a23 and a24 contain nondecomposable SALARY information revealed to users. Similarly, populations a1Ø, a11, a12, a3 and a4 contain nondecomposable ABSENT-DAYS information revealed to users. The intersections of these populations will give SA-populations a17, a18, a19, a2Ø, a21, a22, a13, a14, a15, a16, a23 and a24. Notice that a SA-population may contain one or more A-populations (e.g. a23 and a24).

4.2.5 Security Constraints

Dynamics of the real world or the existence of complex relationships between populations may lead the DBA to impose constraints on the security related information in

populations. The DBA should be able to state the conditions under which any statistical query about any protected property of a population may be reported to users. Since the aim of this section is only to provide the DBA with the power to do so, in general three types of constraints will be distinguished. (Defining these constraints is very much dependent on the specific environment and we are unable to give more detailed analysis and structural specifications of the constraints as done by [Hammer and McLeod, 1975] for semantic integrity constraints).

1) Security Atom Constraints (SA-constraints) apply to the SA-value set in a SA-population A and all populations that contain A. An example is: sum salary information must not include the salary x of employee a in SA-value set w until there is another employee hired or fired.

2) Global constraints (Type 1) apply to the individuals in a population A and individuals of all or some of the populations in the hierarchy that contain A.

Consider Figure 4.4 and the example 4.1 given in the introduction. Assume user group u is allowed to access down to Systems Analyst in the hierarchy. Now the hiring of two new Systems Analysts with Ms and with total salary $27K should not be incorporated into the population Systems Analyst. However if the range of salaries of Computer Scientists with Ms include $14K due to its other child

populations, then the new change may be incorporated into the populations Computer Scientist with Ms and Computer Scientist (if other constraints are also satisfied).

3) Global constraints (Type 2) apply to the individuals of a population A and to individuals of another population B in a different part of the hierarchy.

Example 4.4 Consider the database of employees, projects and assignments in Figure 4.5. It is known that at least ten programmers and one systems analyst with more than ten years working experience are involved in the database development project. Assume the user u also knows that a project leader must be a systems-analyst with PhD. Now if COUNT queries of SYSTEMS-ANALYST-WITH-MORE-THAN-10-YEARS-EXPERIENCE and ASSIGNMENT-IN-DATABASE-PROJECT return 1 and 11, respectively, and the user u knows a systems-analyst x with more than 10 years experience then the user u discloses that x is the project leader of the database development project and also has a PhD. To prevent this disclosure, type 2 global constraints applied to SYSTEMS-ANALYST-WITH-MORE-THAN-10-YEARS-EXPERIENCE and ASSIGNMENT-IN-DATABASE-PROJECT may state that if COUNT information of these two populations are smaller than a1 and 10+a2, respectively, where a1 and a2 are properly chosen small integer constants, then COUNT information of both of the populations are not revealed to users.

Example 4.5 Consider Figure 4.5. Assume two programmers are hired. Now if COUNT information of both PROGRAMMER and ASSIGNMENT-IN-DATABASE-PROJECT increase by two then the new programmers are assigned to the database development project. If this information is to be protected then type 2 global constraints applied to child populations of ASSIGNMENT may state that new assignments in child populations of ASSIGNMENT are reported only when there are new assignments in two or more projects.

4.3 A Statistical Security Management Facility

The formal definition of the SDB in Section 4.2.2 includes type-1 operations for high-level schema modifications and SDB Knowledge Base with its own set of operations. For the simplicity and the efficiency of the design, the SDB design in this section will not include type-1 operations, and a very limited version of the Knowledge Base will be introduced. However, an SDB design that permits type-1 operations and a more general SDB Knowledge Base will be discussed in Chapter 5.

In this section, a statistical security management facility (SSMF) with three principal components is proposed.

(1) A Population Definition Construct (PDC)

(2) A User Knowledge Construct (UKC)

(3) A Constraint Enforcer and Checker (CEC).

The PDC of a population contains information about the

population, related security constraints, changes of the population, etc., in order to achieve effective protection. The UKC of a user group is designed to record users' additional knowledge and SA-constraints. Finally, the CEC consists of several algorithms designed to keep the PDCs and UKCs up-to-date, to enforce the security constraints and to help the DBA in security-related decision problems.

Population Definition Construct:

For each population P, there is one PDC which contains the following information.

(a) Description of the population and its parent, child and sibling populations.

(b) Lowest permissable user group level.

(c) Information as to how changes are included in P.

(d) Allowable statistical query types for each property of P.

(e) Global constraints of P.

(f) If P is an SA-population then description of SA-constraints for each SA-value set of P.

(a),(d) and (f) are self-explanatory.

(b) Assume user groups are classified by levels such that user groups with higher levels have more access power to the database than the user groups with lower levels. Lowest permissable user group level is a level n such that user groups with level m $\geq$ n can access that population.

```
Population PROGRAMMER
     description [Phrase],
     parent populations [COMPUTER-SCIENTIST],
     child populations [(PROGRAMMER-WITH-BS, PROGRAMMER-
          WITH-MS), (PROGRAMMER-WITH-0-4-YEARS-EXPERIENCE,
          PROGRAMMER-WITH-5-10-YEARS-EXPERIENCE,
          PROGRAMMER-WITH-11-OR-MORE-YEARS-EXPERIENCE)],
     other populations in the same cluster [SYSTEMS-
          PROGRAMMER, SYSTEMS-ANALYST],
     lowest permissible user group level 2,
     allowable query COUNT,
     changes processed in PAIRS,
     protected property SALARY,
         allowable query SUM,
     protected property ABSENT-DAYS,
         allowable query MEDIAN,
     global constraints
         constraint 1
              description [Phrase],
              call VIOL-CS1,
         constraint 2
              description [Phrase],
              call VIOL-CS2,
end.
```

Figure 4.7. The PDC of Programmer

(c) Changes due to the dynamics of the real world may be processed in many ways. How these changes are handled is described in the PDC.

(e) Global constraints may be static or dynamic. They may evolve and change as the DBA modifies them, e.g. a manager changes companies, thus extends his knowledge and the DBA should take necessary actions. For each global constraint, the PDC contains the description of the constraint and a call for a routine in the case of violation of the constraint. Figure 4.7 contains the PDC of Programmer in the generic hierarchy described in Figure 4.4.

User Knowledge Construct:

For each user group u, the UKC records the users' additional knowledge about individuals in the SDB. Figure 4.8 contains the UKC of user group u for the generic hierarchy described in Figure 4.4.

Assume user group u is at the 3rd level which can access all populations in the hierarchy.

Users in user group u can identify the individuals that are updated, inserted or deleted from the population PROGRAMMER (e.g. the newly inserted, deleted or updated programmer in the population PROGRAMMER is known by the user group u). For each population, this information is defined after the keyword "identifiable dynamics" in the UKC. Clearly, protection measures to be applied should be

different for a user group which identifies only inserted individuals of a population and a user group which identifies both inserted and deleted individuals of the same population. (There may be other variations, for example, users in user group w may identify updated individuals when the update is from Systems Programmer to Systems Programmer, etc..)

Each SA-population contains one SA-value set for each of its properties. Dynamics of a SA-population (i.e. inserted, deleted, updated individuals) are recorded in a list called the change sequence in the order of occurrences of changes. (This list may be kept separately if the expected number of changes is large). Depending on the type of statistical information revealed, the change sequence is used in several procedures to decide whether the security of individuals and the protected information are in danger.

For security purposes, changes may be processed in groups, say triplets. In such cases, some individuals may be waiting to be processed; these individuals are described and maintained in SA-constraints. For each SA-value set, users may know global upper or lower bounds of the property values of individuals, and upper or lower bounds for some specific individuals. For example, in Figure 4.8, users in user group u know that the salary of programmer Ian Munroe is less than $18K.

```
USER GROUP U [user-id, user-id,...,user-id],
     user group level 3;
     population COMPUTER-SCIENTIST,
         identifiable dynamics INSERTION, DELETION, UPDATE,
     population PROGRAMMER,
         identifiable dynamics INSERTION, DELETION, UPDATE,
     population PROGRAMMER-WITH-Ø-4-YEARS-EXPERIENCE
                        [SA-POPULATION],
        identifiable dynamics INSERTION, DELETION, UPDATE,
        protected property SALARY,
             security atom constraint: {JOHN DOE} is not
                   included,
             change sequence parameters
                active individuals set {(STEVE HART,
                     ROCK HO,2Ø),...,(JOHN GRAY,,3)},
                reachability constant Ø.1,
                largest reachability set size 2Ø,
             known value set {(JOHN SO),....,(ALAN POE)},
             known global upper bound $34K,
             known global lower bound $8K,
             known upper bounds set {(IAN MUNROE,$18K),...,
                   (GEORGE HO,$2ØK)},
             known lower bounds set {Ø},
             change sequence {[(JIM JOE,INSERT),
                   (JACK YU,DELETE),(OLD MEDIAN,$15K),
                   (NEW MEDIAN,$14K)],...,
                   [(PHILIP HO,DELETE),(JACK FU,DELETE),
                   (NEW MEDIAN,$18K)]},
        protected property ABSENT-DAYS,
             security atom constraint: {STEVE HUDSON} is
                   not included,
             change sequence parameters
                active individuals set {(STEVE HART,,15),
                     ...,(JOHN GRAY,,7)},
                reachability constant Ø.2,
                largest reachable set size 15,
             known value set {Ø},
             known global upper bound 9Ø,
             known global lower bound Ø,
             known upper bounds set {Ø},
             known lower bounds set {Ø},
             change sequence {[(JIM JOE,INSERT)],
                  [(JACK YU,DELETE),(CHEN TU,DELETE),
                  (OLD SUM,45Ø),(NEW SUM,345)],...},
     population PROGRAMMER-WITH-5-1Ø-YEARS-EXPERIENCE
                        [SA-POPULATION],
                        .
                        .

end.
```

Figure 4.8. The UKC of user group u for the generic hierarchy described in Figure 4.4.

Change sequence parameters are described as reachability constant w1 and largest reachable set size w2 (see Section 3.3). For each change sequence, an active individuals set is also maintained. (A vertex is inactive if it corresponds to an individual deleted from the population; it is called active if it corresponds to an individual previously inserted but not yet deleted from the population). An active individuals set contains one set element for each connected component with one or two active individuals. Each set element contains the names of the active individuals and the number of vertices in that connected component (i.e. the reachability set size). For example, in Figure 4.8, the information graph of the population of Programmer with 0-4 years of experience contains a connected component with two active vertices for individuals Steve Hart and Rock Ho, and the number of vertices in that connected component is 20.

Constraint Enforcer and Checker:

The CEC is composed of several algorithms. It utilizes PDCs and UKCs to perform the following two basic tasks:

(a) For each statistical query, it is invoked to find out the global and SA-constraints by tracing the related PDC and the UKC, to enforce these constraints by executing the related procedures (and thus altering the answer to the user's statistical query, if necessary);

(b) For each change (i.e. insertion, deletion or update of individuals) in the populations of the D-A Model, it is invoked to modify the constraints, to decide whether to process (i.e. to include into users' statistical queries) the change for each SA-value set, and (if the change is processed) to modify the related change sequence and its parameters for each user group u.

Other than above, the CEC helps the DBA in several security-related decision problems by providing lists of individuals whose security is threatened under events described below.

(1) Changes in user groups. User groups may join, decompose, or users may move from one user group to another. In these cases, additional knowledge of a user group may increase, and further disclosed information is then decided by the CEC using the UKC of the user group and change sequences.

(2) Changes in the conceptual model such as decomposing a population or re-partitioning a population. In these cases, the CEC finds SA- and A-populations, re-arranges UKCs, modifies security measures and reports disclosures.

(3) Changes in users' additional knowledge such as a modified known value set or an updated known upper bounds

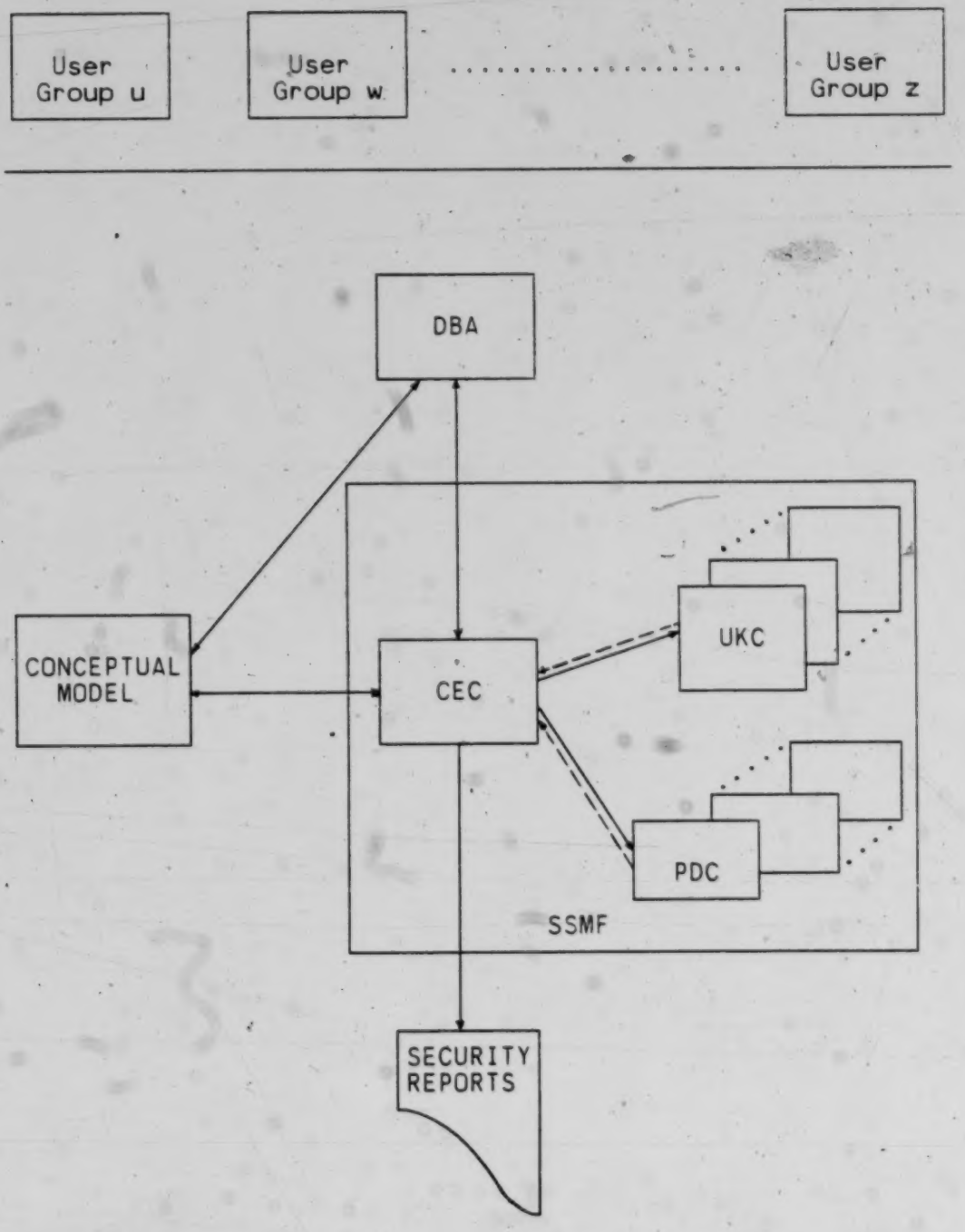


Figure 4.9 Statistical Database Model

set. In these cases, further disclosures are decided by tracing the change sequences and considering possible inferences discussed in Section 4.4.

Another job of the CEC is to modify and maintain several security measures related to the change sequence of each SA-value set in order to give the DBA a measure of how secure the system really is at a particular time. Some of these measures are discussed in Section 3.3.

The general scheme of the SSMF is depicted in Figure 4.9. The CEC utilizes the conceptual model, PDCs and UKCs to enforce security constraints and modify statistical queries. Individual insertions, deletions and updates into populations in the conceptual model are intercepted, and modifications of security constraints, change sequences and their parameters are done by the CEC. In the case of disclosures, the DBA is notified, and security-related reports such as the values of security measures, the number of constraints in effect, the number of individuals not included into statistical queries and the introduced error, etc., are reported by the CEC.

4.3.1 Implementation Considerations

In this section, implementation and maintenance issues of security-related structures are discussed.

To answer a statistical query, conventional database

query processing is performed to obtain the answer, and then related security constraints (i.e. global and SA-constraints) are enforced. To retrieve SA-constraints, the user group of the user is determined and the related UKC is accessed. To retrieve global constraints, the PDC of the population described in the user's query is accessed. Thus the additional time overhead to the conventional database query processing involves two accesses and the processing of security constraint routines.

To process changes (i.e. insertion, deletion or updates of individuals), in addition to the conventional database query processing, the related PDC is accessed to modify global constraints, and then, for each user group, the UKC is retrieved to update SA-constraints, the user group's knowledge and change sequence and its parameters.

Below possible operations on the Known Value Set (KVS), the Known Upper and Lower Bounds Sets (KUBS and KLBS), change sequences and their parameters in the UKC are discussed. The following notation is used for an SA-population with m individuals and p properties. The i^{th} property value of j^{th} individual in the population is denoted by p_{ij}. The KUBS of the i^{th} property for the user group G is denoted by KUBS(i,G). v_{ij} is the known upper bound value for the i^{th} property of individual j in the SA-population. v_{ij} is either an element of KUBS(i,G) or equivalent to the known global upper bound for the i^{th} property.

The operations on the KVS are set membership check, insertion and deletion. Clearly, the KVS is a dictionary [Aho et al., 1974]. Thus a balanced tree (a 2-3 tree or an AVL tree) may be used to process an operation of the KVS in $O(\log_2 m)^+$ time, where m is the number of individuals in the population.

The KUBS(i,G) should support the following operations:

a) Search for

   1) v_{ij} for a given j,

   2) all j such that $v_{ij}-p_{ij} \leq c$, c is a constant,

b) insertion of v_{ij} into KUBS(i,G) for a given j,

c) deletion of v_{ij} from KUBS(i,G) for a given j,

d) update of v_{ij} for a given j.

Operation a-2 may be required to assess how "close" the users' knowledge of the upper bound value v_{ij} to p_{ij} is. The operations on the KLBS are similar.

For the KUBS(i,G), operation d is equivalent to two operations, operations b and c. Thus, operations a-1, b, c and d imply a dictionary, and the balanced tree T_1 with each leaf node j (j is an individual in the population) containing $v_{ij} \in$ KUBS(i,G) may be used. For operation a-2, if it is executed rarely then a sequential search is sufficient otherwise another balanced tree T_2 (a 2-3 tree or an AVL tree) may be used, whose external node is a linked-

$^+$O-notation is described in [Aho et al., 1974].

list containing individuals j with the same $(v_{ij}-p_{ij})$ values , and each external node is linked to the external node to its right. Thus, if there are n_i different $(v_{ij}-p_{ij})$ values in the SA-population, operation a-2 requesting all j such that $(v_{ij}-p_{ij}) \leq c$ requires $O(\log_2 n_i)$ comparisons for determining the external node e containing individuals j with $(v_{ij}-p_{ij})=c$, and then sequential traversal of the external nodes to the left of the external node e is sufficient to complete operation a-2. If each external node i of T_1 containing $v_{ij} \in KUBS(i,G)$ also contains a pointer to the individual j's corresponding node in T_2 then maintenance of T_2 due to operations b, c and d take $O(\log_2 n_i)$, $O(1)$ and $O(\log_2 n_i)$ time, respectively. Thus operations a-1, a-2, b, c and d take $O(\log_2 m)$ time.

The change sequence is consulted during checking for disclosure, and modified during a change in the population. The operations on the change sequence are:

1) inserting a new change into the change sequence and modifying the change sequence parameters,

2) finding a certain reachability set from the change sequence,

3) finding all reachability sets from the change sequence.

The change sequence may be implemented as a list. Including a new change into the change sequence can then be achieved by simply inserting a node at the front of the

list. To modify the active individuals set, a search in the active individuals set is needed for each deleted individual. If there are few active individuals then a linear search is sufficient. In the case of large number of active individuals, this search may be faster by implementing a dictionary, e.g. by keeping the active individuals in the leaves of the balanced tree T_3. Each active individual in T_3 contains a pointer to the other active individual in the same connected component of the information graph. Thus maintenance of the active individuals set can be performed in $O(\log_2 k)$ time, where k is the number of individuals in the active individuals set. The reachability constant w_1 can be updated without extra time complexity if each active individual in the leaves of T_3 also contains the name of its reachability set. For the largest reachable set size w_2, maintaining the name of the largest reachable set and keeping in each leaf node of T_3 the number of individuals in the related reachability set is sufficient to maintain w_2 without any extra time complexity. Thus operation 1 can be processed in $O(\log_2 k)$ time, where k is the size of the active individuals set.

All reachability sets of an SA-population may be found by a single sequential backward traversal of the change sequence as follows:

(i) Create one reachability set Rs_i for each element in the active individuals set. Also for each active individuals

set element having a single active individual, r_a, or two active individuals, r_b and r_c, create an element for a set A (initially empty) as $(r_a,,Rs_i)$ or (r_b,r_c,Rs_j) respectively, where Rs_i and Rs_j are the related reachability sets.

(ii) Traverse the change sequence backwards and for each change sequence tuple, (r_k, r_t), perform a, b and c below and stop.

(a) If each of r_k, r_t is not in any 3-tuple x of A, create a new reachability set Rs_i with individuals r_k, r_t, and create the set element (r_k, r_t, Rs_i) for A.

(b) If only $r_k(r_t)$ is in a 3-tuple x of A then replace $r_k(r_t)$ with $r_t(r_k)$ in x. Add $r_t(r_k)$ to the reachability set specified in the 3-tuple x.

(c) If both r_k and r_t are in a 3-tuple x of A then remove x from A.

If each external node in T_3 is linked to the external node to its right, then step i can be performed by scanning the active individuals set in O(k) time, where k is the number of individuals in the active individuals set. Set A can be implemented as a dictionary. For example, a balanced tree S whose leaf node containing an individual in a 3-tuple of A , related reachability set and a pointer to the other individual in the 3-tuple may be used. Then step (ii) takes $O(t\log_2 s)$ time where t is the number of tuples in the change sequence and s is the maximum size of A at any time. Thus operation 3 takes $O(k+t\log_2 s)$ time.

Another implementation issue is a fast access to all A-populations of a given population, which is needed to enforce SA-constraints. This problem requires a traversal in the related UKC. Note that UKCs are expected to be very large data structures. In the D-A model, this problem takes the form of hierarchical traversal. Since the UKC is very large, one may assume that it resides in the secondary memory. Each user query triggers a traversal in the UKC, and in a paging environment, there are page defaults. The aim is to minimize these defaults.

It may be assumed that the population specified by the query is accessed using a table or, for example, using base displacement. Thus it is sufficient to consider only the problem of finding its A-populations. Clearly, given a population, there are several paths to access its A-populations since a node in the UKC may have several clusters. Thus access paths in the UKC which require minimum number of node visits must be determined first. This can be performed in O(n) time, where n is the number of nodes in the hierarchy, by a simple algorithm as follows: (a) start from level k, k=(highest level in the hierarchy)-1, (b) for each node at level k, find its cluster to be used in the access path. Record the "number of edges to be traversed", (c) decrease k by 1. If k>1, go to (a) otherwise stop.

Assuming the frequency of access of each node v_i in the

hierarchy is $R(v_i)$ and P is the page size, the problem now becomes optimum partitioning of the graph formed by access paths. This problem is known to be NP-complete [Garey and Johnson, 1979]. There are however some heuristics which require $O(n^2)$ or $O(n^2 \log_2 n)$ time [Kernighan and Lin, 1970]. One approach is to change the access path graph into a tree by an approximation algorithm in $O(n)$ time, and then find the optimum partitioning of the tree using dynamic programming approach in $O(nP^2)$ time [Lukes, 1974].

4.4 Protection Requirements for Different Statistical Queries

In this section possible security constraints for different statistical query types are investigated. First, inferences available to users are identified, then related security constraints to enforce security are briefly described. One distinguishes three different types of inferences by users.

(a) Type S inferences due to the hierarchical structure of the conceptual model.

(b) Type D inferences due to the dynamics of the real world.

(c) Type R inferences due to existing relationships between individuals in different populations or in the same population.

Disclosures in examples 4.1, 4.4 and 4.5 are due to

type R inferences. Type R inferences are dependent on the specific environment. In this chapter, it is assumed that global constraints are defined by the DBA to prevent disclosures due to type R inferences. That is, the DBA is responsible for identifying type R inferences and applying protection measures to prevent compromise. Another approach may be to define formally type R inferences and use a theorem prover to decide about the inferred knowledge and the disclosed information. In Chapter 5, this approach is outlined, which uses a Question-Answering System to enhance the security of the SDB.

Below only type S and type D inferences are considered. The following are suggested schemes for a sample of different types of statistical queries; the others can be derived similarly.

4.4.1 COUNT Queries

Assume only COUNT queries are allowed and individuals in populations are identifiable. Assume Systems Analyst with Bs is decomposed into two subpopulations as "Systems Analyst with Bs and convicted of felony" and "Systems Analyst with Bs and not convicted of felony". It is well known [Hoffman and Miller, 1970] that

```
COUNT(Systems Analyst with Bs)=COUNT(Systems Analyst with Bs
                                     and convicted of felony

John Doe is a Systems Analyst with Bs
```

⟶ John Doe is convicted of felony.+

Similarly,

[COUNT(Systems Analyst with Bs and not convicted of felony)=0
  John Doe is a Systems Analyst with Bs] ⟶ John Doe is convicted of felony.

Thus for type S inferences we need the following global constraints.

(a) in population SYST-ANALYST-BS-CONV-FELONY,

If COUNT(any superpopulation of SYST-ANALYST-BS-CONV-FELONY) - COUNT(SYST-ANALYST-BS-CONV-FELONY) ≤ a
then individuals creating above difference are not reported in COUNT queries.

(b) in population SYST-ANALYST-BS-NOT-CONV-FELONY,

If COUNT(SYST-ANALYST-BS-NOT-CONV-FELONY) ≤ a
then COUNT(SYST-ANALYST-BS-NOT-CONV-FELONY) is not answered.

The above disclosure type and constraint have been proposed and discussed widely in recent studies [Schlorer, 1975; Schlorer, 1976; Denning et al., 1979]. If introduced in a controlled environment, it is a viable protection procedure.

The constraints described above can be easily modified to consider the users' knowledge described in the UKC by changing a to (a+x) where x is the size of the user group's knowledge in the population.

Type D inferences may be avoided likewise, e.g. only

+ ⟶ means "implies" or "imply"

when there are (a1+x) insertions or (a2+x) deletions from either populations of SYST-ANALYST-BS-CONV-FELONY or SYST-ANALYST-BS-NOT-CONV-FELONY then changes are reported to user group u where insertion and deletion of SYST-ANALYST-BS are identifiable.

4.4.2 SUM and COUNT Queries

Consider Figure 4.4 and assume only SUM and COUNT queries are allowed for all populations. Assume further that users in user group u can always identify if a programmer is hired or fired but cannot identify if he is a systems programmer or a systems analyst. Now any changes in populations Systems Analyst or Systems Programmer can be reported to user group u immediately but care must be taken in reporting changes in the population Programmer. In what follows it is assumed that insertions and deletions into a population are identifiable and only type D inferences are considered. Later type S inferences are also discussed.

Using the information graph, it is shown in Chapter 3 that there are several ways of securing the SDB for SUM and COUNT queries. Clearly, these results should incorporate users' knowledge in order to be meaningful. If individuals with known (or suspected) property values are processed in pairs only with known individuals then the SDB will still be secure, since unknown values will be separated from known values in the equations in pairs. Thus in order to prevent

disclosures due to type D inferences, we have

(a) populations with even number of individuals, and

(b) for each SA-value set in each UKC, there is a SA-constraint which delays the processing of (known and/or unknown) individuals with recent changes until another change occurs. (see SA-constraint in Figure 4.9.)

Clearly, if each population has an even number of individuals then the size difference between a population and any of its parent populations must be either zero or at least two. Thus type S inferences cannot cause any disclosure.

A population A may contain several SA-populations and thus several individuals may be waiting for inclusion in statistical answers of A. To prevent that, when t unknown (known) individuals are waiting then they may be included in the statistical answers of A. This can be specified by global constraints in A and its parent populations. The bound t must be large enough so that the information revealed to users will be practically useless for all disclosure purposes.

4.4.3 MEDIAN and COUNT Queries

Assume only MEDIAN and COUNT queries are allowable for all populations in the conceptual data model. Both type S and type D inferences may lead users to obtain upper or lower bounds for protected property values if insertions,

deletions, or updates are identifiable. The following examples illustrate the possible inferences.

Example 4.6 (type S inference)

Consider Figure 4.4. Assume individuals are identifiable and there is only one Programmer with Ms whose salary is x. Thus queries about Programmer with Ms are not permitted. However, the following information is obtainable.

MEDIAN(PROGRAMMER-BS,SALARY)=a MEDIAN(PROGRAMMER,SALARY)=a1

COUNT(PROGRAMMER-BS)=n COUNT(PROGRAMMER)=n+1

Now, $a1 > a \longrightarrow x > a$ and $a1 < a \longrightarrow x < a$

Thus one has an inference about the salary x of the only programmer with Ms.

Example 4.7 (type D inference)

Consider Figure 4.4 and assume changes are identifiable.

MEDIAN(SYSTEMS-ANALYST-WITH-BS,SALARY)=a

COUNT(SYSTEMS-ANALYST-WITH-BS)=m

Assume a new Systems Analyst with Bs and salary x is hired:

MEDIAN(SYSTEMS-ANALYST-WITH-BS,SALARY)=a1

COUNT(SYSTEMS-ANALYST-WITH-BS)=m+1

Now $a1 > a \longrightarrow x > a$ and $a1 < a \longrightarrow x < a$

We would like to avoid above inferences.

Processing Changes in Pairs:

First type D inferences will be considered then type S inferences will be discussed. Consider population A with protected property B and assume all changes are processed in pairs.

MEDIAN(A,B)=a COUNT(A)=n

Two individuals with property B values x and y are added to A, then

MEDIAN(A,B)=a1 COUNT(A)=n+2

Now (a) n is odd

$a1 < a \longrightarrow x, y \leq a1$

$a1 > a \longrightarrow x, y \geq a1$

(b) n is even

$a1 < a \longrightarrow$ at least one of $x, y \leq a1$

$a1 > a \longrightarrow$ at least one of $x, y \geq a1$

Clearly, (b) is better than (a) in the sense that it does not allow an upper or lower bound inference for any of the property values x, y.

Consider now

$$\left.\begin{array}{l} \text{MEDIAN}(A,B)=a \\ \text{COUNT}(A)=n \end{array}\right\} \begin{array}{l} \text{an individual with property x is added to A} \\ > \\ \text{an individual with property y is deleted from A} \end{array} \left\{\begin{array}{l} \text{MEDIAN}(A,B)=a1 \\ \text{COUNT}(A)=n \end{array}\right.$$

and we have the following inferences

(a) n is odd

$al<a \longrightarrow (x \leq al)\&(y \geq a)$

$al>a \longrightarrow (x \geq al)\&(y \leq a)$

(b) n is even

$al<a \longrightarrow x < y$

$al>a \longrightarrow x > y$

Thus if every population always has an even number of individuals then processing changes in pairs prevents any direct inference of the property values of individuals. One can also use this result for type S inferences by having the global constraint that the difference in size between any population and its parent population should at least be two. This requirement actually is always satisfied if all populations start with an even number of individuals.

Processing Changes in Triplets:

Having populations with an even number of individuals and processing changes in pairs still has the following deficiency. Let the median value a be the average of two protected property values u and v, $u \leq v$; and two individuals with protected property values x and y are added into the population, then

$x,y \notin [u,v]$ & $(al<a) \longrightarrow x,y<al$

and $x,y \notin [u,v]$ & $(al>a) \longrightarrow x,y>al$.

Thus for large population size n and $al<a$ it is highly probable that $x,y<al$ or similarly for large n and $al>a$, we have $x,y>al$ with high probability. If this deficiency and the requirement of even population size are not tolerable

then changes are processed in triplets. The following inferences are possible.

Assume MEDIAN(A,B)=a and COUNT(A)=n.

(a) individuals with property values x,y,z are added to population A.

MEDIAN(A,B)=a1 COUNT(A)=n+3

For even or odd n,

$a1 < a \longrightarrow$ at least two of $x,y,z \leq a1$

$a1 > a \longrightarrow$ at least two of $x,y,z \geq a1$

(b) an individual with property value x is deleted from A and individuals with property values y,z are added to A.

MEDIAN(A,B)=a1 COUNT(A)=n+1

For even or odd n,

$a1 < a \longrightarrow$ at least one of $y,z \leq a1$

$a1 > a \longrightarrow$ at least one of $y,z \geq a1$

Other changes (i.e. one addition and two deletions or three deletions) result in similar inferences.

Similarly, to prevent disclosures due to type S inferences one can have a global constraint that the size difference between any population and its parent population should at least be three or the individuals creating this difference are not reported. Also hierarchical structure of the conceptual model should be taken into account for type D inferences. Consider Figure 4.4, assume three Systems Analysts with Bs and with salaries x,y,z are hired and median salaries of populations Systems Analyst with Bs,

Systems Analyst and Computer Scientist have changed from a1,a2,a3 to b1,b2,b3 where b1<a1, b2<a2, b3<a3. Now if the DBA learns that user group u had in fact known the value of $z > \max\{a1,a2,a3\}$ then $x,y \leq \min\{b1,b2,b3\}$ is revealed. Thus this inference should be recorded into the UKC of user group u.

Changes (whether processed in pairs or triplets) should be recorded into the related change sequence for auditing and for other tasks of the CEC described in Section 4.3.3. Also individuals with known (or suspected) property values should be processed only with known individuals to avoid direct inferences about protected property values.

CHAPTER 5

EXTENSIONS FOR A SECURE SDB DESIGN

Extensions to the protection scheme of employing constraints in the conceptual data model are discussed in order to increase the effectiveness of the protection scheme and the richness of the SDB. It is argued that a Question-Answering System with deductive inference mechanisms may be very useful for the database system in deciding inferred knowledge. A set of security-related commands for controlled changes about individuals and populations in the conceptual data model are proposed. The benefits of a security kernel architecture for the SDB are discussed. Finally, a complete SDB design is discussed, which includes protection data, Question-Answering System, security-related commands and a security kernel.

5.1 Introduction

This section discusses possible extensions to the SDB design in Chapter 4.

5.1.1 Inferred Knowledge and a Question-Answering System

In Chapter 4, inferences due to existing relationships

in the real world environment (i.e. type R inferences) are controlled in an ad hoc manner. That is, the DBA is responsible for identifying these general rules and applying security measures to prevent compromise. Moreover the SDB design in Chapter 4 does not maintain users' knowledge of general rules. The UKC contains a part of users' knowledge, namely, the protected property values of individuals. Although the SDB Knowledge Set defined in Section 4.2.2 contains users' knowledge relevant to the application, the UKC does not contain users' knowledge of

(a) explicit facts that are not represented in the conceptual model which may be utilized for disclosure, e.g. "individual a is the brother of individual b", and

(b) general rules that may lead to disclosure, e.g.

(1) Income of a and b is x.

(2) b is not working and has no extra income.

(3) b is the ex-wife of a.

(4) Every nonworking ex-wife gets w% of the income of ex-husband.

⟩ Income of a=100x/(100+w) and Income of b=wx/(100+w)

Global security constraints are proposed in Chapter 4 to avoid this kind of compromise. However, facts (2) and (3) may not exist in the conceptual model (they are perhaps only needed for security purposes), and general rule (4) may (or may not) be known to some users. Clearly, a static global constraint may fall short for security depending on particular users' additional knowledge. For example, a

certain user equipped with other explicit facts and general rules may be able to deduce more. A better approach may be the formal treatment of the question "what can be inferred with certain knowledge?". (It is assumed that the database system is correctly informed of each user's knowledge).

In order to maintain users' additional knowledge relevant to the security of the SDB, and make decisions about the implicit deducible information, the following extensions are now proposed.

(1) As described in Section 4.2.2, a Knowledge Set for each user group, containing security-relevant general rules and explicit facts that are known by the users and are not described in the conceptual model, and

(2) a Question-Answering System (QAS) with a deductive capability. Notice that the Knowledge Sets and the QAS will only be used by the DBA or the security-related procedures. Advantages of adding the QAS with deductive capability include the following:

(a) The QAS may help the DBA in the problems concerning security-related decisions by answering conditional questions. For example, "Is the salary of dataperson A deducible by user group u if A is assigned to project p?" may be answered by the QAS as "yes, if he is assigned as a manager or if he is an engineer with a Ph.D.". The QAS can generate answers for the DBA to those questions

asking whether compromise may occur (or has occurred) during events such as changes in the database (say, insertion of individuals), changes in the conceptual model (say, new populations) or changes in the knowledge set of users (say, learning a new general rule). If some of the actions to these events are pre-specified (such as enforcing a certain constraint), then these events may be automatically processed by security-related procedures without consulting the DBA.

(b) The QAS may return the reasoning for its answers about the security of information in the SDB. This helps the DBA decide how and where to apply security constraints effectively without unnecessarily reducing the richness of the database. Also, even in cases when the QAS is unable to give a definite answer, its line of reasoning may help the DBA to decide about the answer.

Since type S and type D inferences may be pre-investigated without using the QAS, analysis and prevention of type S and type D inferences can be dealt separately so as to increase the efficiency of the protection and the QAS may be used only when most needed.

Several QASs have been reported in the literature [Green, 1969; Minker et. al., 1973]. Mostly, the resolution [Robinson, 1965] technique with several improved search strategies [Chang and Lee, 1973; Nilsson, 1971] has been

used to derive answers. However, lately, Natural-Deduction systems [Bledsoe and Bruell, 1974] have been offered as alternatives to resolution.

The idea of incorporating a deductive capability into a database system is not new. Kellogg et. al. [Kellogg et. al., 1976] reports the implementation of a relational database with a deductive processor and general rules file. Minker [Minker, 1978] proposes an inferential system in which there are explicit facts (i.e. extensional database) and general rules (i.e. intensional database). The general rules and explicit facts are used to derive implicit facts within the system. For relational databases with large sets of explicit facts and relatively small sets of inference rules, Reiter [Reiter, 1978] proposes a theorem prover which only looks at the intensional database. The theorem prover's output, which is a set of queries, is then extensionally evaluated.

For the SDB security, the main concern is to decide whether or not a property value of a single individual is compromised. The deductive search for an answer to this question is likely to deal with very few explicit facts. Moreover, the number of general rules relevant to the deductive process of answering a particular security question is usually small. Aiding the deductive search with semantic information may be very useful in increasing the efficiency of the QAS. To state explicit facts and general

rules, 1st order predicate calculus seems to be generally sufficient. In those cases where higher order general rules in the real world exist, the direct intervention and analysis of the DBA are assumed. Below, the input and the output of the QAS are briefly described.

Using the definition of SDB security in Section 4.2.2., the database system is expected to protect confidential facts and some nonconfidential facts. This protection can be reduced to the protection of property values of individuals in the SDB. The fact that individual a has the property value s can be represented by the predicate P(a,s). When the DBA wants to check whether user u can disclose the property value s of individual a, he asks the question Q: Ex P(a,x)[+] together with a pointer to the Knowledge Set of user u. In other words, The DBA wants to prove that the well-formed formula Ex P(a,x) is valid. If Q is valid then as the answer, "yes" and a term[++] are returned. If Q is not valid then there are two possibilities; either the inference

---

[+]Ex means "there exists x".

[++]The answer can be (i) a term involving only known (non-skolem) constants, functions, and variables, or (ii) a term involving a skolem constant or function. In case (i), u can compromise the value s. In case (ii), the x in question is known to exist, but because of the involvement of skolem constant or functions, its value remains unknown. If the user u's knowledge of the existence of s is not desirable by the system then the answer "yes" will protect this information. Otherwise, the answer "yes" may be replaced with "no, insufficient information due to a skolem (cont'd)

procedures run out of time, space or some other criteria in which case they return "cannot be answered due to search limitations" or the inference procedures come to a halt in searching relevant inferences in which case the answer returned is "no due to insufficient information". The latter answer simply says that the system cannot infer that there exists x with P(a,x) using the "given" Knowledge Set of the user u and hence, nor can the user u. Since the DBA is only interested in whether or not user u can infer P(a,s), for our purposes, a "no due to insufficient information" answer implies the security of the protected value s. If the QAS returns "cannot be answered due to search limitations", either the DBA is notified or the search is re-tried with different parameters.

5.1.2 Changes and Adaptability of the Conceptual Model

It is generally agreed [Weber, 1976; Winograd, 1973] that a static representation of knowledge is insufficient to model the real world correctly. Moreover completeness of a conceptual model is dependent on (a) the ability of the designers who create the model and (b) the purpose the model serves [Weber, 1976]. Therefore every aspect of the

---

function". Certainly the answer "no" may unnecessarily restrict the usefulness of the QAS. Perhaps a change in the theorem prover may result with case (i) as an answer rather than case (ii). More research is needed to eliminate case (ii) as the answer to the question.

conceptual data model should have dynamic capabilities, i.e. populations may be formed, deleted or decomposed; individuals may be inserted, deleted or (their properties) updated; relationships may be added or deleted from the conceptual model. For the SDB, these capabilities are procedurally described by the Operation Types Set in Section 4.2.2 and are of utmost importance since users do not have high level data manipulation operators. Note, however, that changes to the SDB data model should either be extensions or reflect changes in the real world environment [Date, 1977]. Moreover, these changes have to be made only by the DBA.

The conceptual model of a general-purpose database is defined by means of the conceptual schema which is represented by a specially provided language, perhaps a data definition language [Date, 1977]. The compiled form of the conceptual schema is used by the database management system and the source form serves as a reference document. For a general-purpose database, changes to the conceptual model can be achieved by re-writing a new conceptual schema and then compiling it. However, for the SDB schema (defined in Section 4.2.2), (a) structural rules, if any, have to be satisfied (e.g. rule 1 in Section 4.2.3), (b) the danger of compromise due to changes has to be checked, (c) users' additional knowledge has to be recorded in the UKCs and Knowledge Sets, (d) PDCs for new populations have to be constructed, and (e) security constraints related with the

changes have to be specified. It is therefore desirable to have a set of high-level operations (or commands) for changes in the SDB schema. These operations are procedurally defined by type-2 operation types in SDB data model and Knowledge Base Operation Types Set in Section 4.2.2.

The job of the SSMF can be extended to check for compromise by consulting the QAS and to process (c)-(e). One important advantage of these additions to the SDB is to ease the job of the DBA by deferring some of his duties to the SSMF.

5.1.3 Security Kernel for the SDB

Certification of the security mechanism of the SDB is needed to guarantee that it works correctly. Since certification by proving the correctness of programs is known to be a difficult task, it is desirable to minimize the software needed for certification.

Security kernels have been successfully applied in operating systems in order to improve the reliability of the system. Recently ([Downs and Popek, 1977], [Downs and Popek, 1979]) have reported a general design for a secure database management system (DBMS), together with a case study implementation using INGRES relational DBMS. The design supports data security through the use of a kernel architecture which minimizes and encapsulates the software upon which correct protection enforcement depends. The main

advantage of a security kernel is to reduce the security relevant code for certification.

In statistical databases, a security kernel can be designed by having certified separate modules containing all security relevant code. The security kernel design in Section 5.3 may be considered as the extension of [Downs and Popek, 1977] to statistical databases.

5.2 The Extended SDB Design

In this section, an SDB design to include the considerations in Section 5.1 is discussed. To illustrate the design, the D-A Model is used as the conceptual data model of the SDB. However the design is independent of the choice of data model and, may easily be modified for any other structured, redundant and semantic data model.

Some users of the SDB have statistical access to the database as well as primitive change operations such as retrieval, insertion, deletion or updates of some individuals in the SDB. For example, managers may change the information about employees in their own department, or an employee may retrieve and change his own information in the SDB. In order to avoid simple direct disclosures, users may be permitted to insert or delete any individual only once, and changes are proposed to be revealed to users in pairs, triplets, etc. (See Chapter 3). It is also assumed that different views of the data and high-level data manipulation

operators such as join, project operators in the relational model are not permitted to users for statistical queries.

Clearly, letting users perform controlled insertions, deletions and updates makes the authorization and enforcement of authorization rules a formidable task. However, in the USA, for example, laws concerning privacy require [Privacy Act, 1974; Davida et. al., 1978] that

(1) a means must be provided for an individual to review his own information in database and how it is used,

(2) the individual must be provided with a means of correcting or amending his own identifiable information. In the SDB design, it is assumed that authorization and enforcement of authorization rules are achieved using mechanisms similar to one of [Hartson and Hsiao, 1976; Fernandez et. al., 1976; Griffits and Wade, 1976] and will not be discussed here.

In the SDB, users and the DBA deal with logical objects, and security constraints are defined over logical individual objects (or individuals) in the conceptual data model. Since the enforcement of security constraints (and authorization rules) are most reliably done at access time on physical objects, a secure mapping of physical and logical individual objects is needed for a secure SDB. In [Downs and Popek, 1977] this mapping is achieved by using tags on each separately protectable data in the physical

database. In this design the same mechanism is used, and each property value of an individual in the physical database is attached with its logical name and the logical name of its property.

The DBA and the SSMF have access to the protection data which is described in Section 5.2.1. Section 5.2.2 contains descriptions of the (extended) SSMF and the QAS. The DBA can insert, delete or update (properties of) individuals in the SDB, make changes in the conceptual model and use other security-related commands to insure the security of the SDB. Section 5.2.3 discusses the set of security-related commands available to the DBA. Section 5.3 discusses how statistical queries, insertion,deletion and update queries of users as well as commands available to the DBA are processed.

5.2.1 The Protection Data

The protection data contains the following.

1) A Knowledge Set which contains a user group's additional knowledge of general rules and explicit facts and a User Knowledge Construct (UKC) are needed for each user group. General rules and explicit facts are properly expressed first order predicate calculus formulas. The UKC contains SA-constraints, some global constraints, users' knowledge of the protected property values in SA-populations, change sequences and change sequence parameters. Note that now, unlike in Chapter 4, the UKC may

contain some global constraints to avoid some type S and type D inferences. The reason for this change is to enable the DBA to apply different constraints to different users so that the richness of the SDB may be less restricted.

2) A Population Definition Construct (PDC) for each population.

3) Logical Name Tables of individual objects in the SDB. To achieve correct mapping between physical and logical individual objects, the logical name tables of individuals are maintained.

4) Authorization Information.

5.2.2 The (Extended) SSMF and the QAS

The (Extended) SSMF consists of three certified modules: the Query Controller (QC), the Constraint Enforcer and Checker (CEC) and the Conceptual Model Modifier (CMM). All three modules of the SSMF have access to the protection data. The Question-Answering System (QAS) is not certified, but works isolated from users, has only read access to the protection data and is used only by the SSMF and the DBA. Similar to [Downs and Popek, 1977], the security unrelated portions of the database management system are separated and put into another module: the Database Management Module (DBMM).

Statistical queries of users should specify the

population's name (e.g. EMPLOYEE) and property (e.g. SALARY), and the statistical query type(s) (e.g. SUM). These queries are called population-specified queries. Another variation to population-specified queries is by specifying the characteristics of the population using conjunctions of boolean clauses instead of the population's name. These queries are called characteristic-specified queries. Functions of the QC include (a) parsing the query and (b) mapping characteristic-specified populations into existing populations in the conceptual model.

Function of the CMM is to help the DBA to assess possible inferences when there are changes in the conceptual model (i.e. when using conceptual model modification commands), and the CMM uses the QAS for deciding about the security of individuals.

The CEC utilizes the PDC, the UKC and the QAS to enforce or to modify security constraints. The CEC also helps the DBA in several security-related decision problems by providing lists of individuals whose security are threatened under events such as changes in user groups or in users' additional knowledge.

The QAS is invoked when (a) users (or the DBA) request insertion, deletion or updates of individuals, (b) users gain "some more" knowledge and further inferences are to be decided, or (c) the DBA wants to change the conceptual model

and the QAS is called through the CMM. In all these cases the function of the QAS is to decide about the inferred knowledge.

5.2.3 Conceptual Model Modification and Security-Related Commands

Conceptual Model Modification commands are used for creation, deletion or decomposition of populations and for attribute deletion or addition to populations. It is equally important to have commands for insertions or deletions of users' additional knowledge as well as for changing security constraints and users' allowed statistical query types. All commands can only be used by the DBA. All commands have a test mode in which case the operation is not executed, but the consequences if it were executed are reported to the DBA (e.g. possible disclosures if the modification took place are reported, etc.). The test mode may be identified by "*" preceding the command, e.g. *CREATE, *DECOMPOSE, etc.. Below we describe these commands based on the D-A Model. However, they may be modified slightly for the use of other semantic data models. Note that while processing these commands, the rule (1) for the D-A model (Section 4.2.3) is not checked by the SSMF, but is confirmed by the DBA.

Population Formation in a New Generalization Plane:

Create population command in Figure 5.1 creates a new population EMPLOYEE in a new generalization plane. Population EMPLOYEE has only one property, SALARY, for which SUM, MEDIAN and COUNT queries are allowed. IND is a binary relation (previously prepared by the DBA) with one tuple for each EMPLOYEE individual, i.e. (empname,salary). EMPLOYEE object is not an aggregate of other abstract objects (thus the keyword aggregate of may be deleted).

Security information gives security-related information. INS-DEL is a relation with one tuple for each user, i.e. (userid,id-ins,id-del), where id-ins and id-del describe whether insertion or deletion of individuals into EMPLOYEE population are identifiable (that is, whether the user group can identify the deleted or inserted individual). SAL-UPDT is a binary relation with a tuple (userid,id-upd) for each user group, where id-upd describes whether salary updates can be identifiable.

FACT is a binary relation with tuples (userid,fcts), where fcts describes an explicit fact known by the user group and not existing in the conceptual model or in the protection data. Similarly, binary relations INFRNCES and VSET describe general rules involving the EMPLOYEE population and SALARY values of EMPLOYEE individuals that are known to user groups. Users' knowledge of upper and

```
create population EMPLOYEE,
    properties SALARY(SUM,MEDIAN,COUNT),
    individuals IND,
    aggregate of,
    security information
      identifiability
        insertion-deletion INS-DEL,
        property-update SALARY(SAL-UPDT),
      knowledge
        facts FACT,
        general rules INFRNCES,
        known-value set SALARY(VSET),
      global constraints,
      changes processed in PAIRS,
      disclosure check EMPLOYEE(SALARY),
end.
```

Figure 5.1. The Create Population Command

lower bounds about employees' salaries may also be specified similarly if deemed necessary.

There are no global constraints for EMPLOYEE population (hence the keyword global constraints may be deleted). If there were, procedure names would be given. Finally the command also specifies that changes (i.e. insertions, deletions and updates) in EMPLOYEE population are processed in pairs in order to prevent disclosure.

Execution of the create population command includes the following.

(1) Conceptual model is modified.

(2) Protection data is modified, i.e. a new PDC is created, the UKC and knowledge base of each user group are updated, logical name tables of individuals are modified. (We assume that the authorization information is specified by other means).

(3) The CMM using the QAS checks further disclosures (e.g. SALARY property values of all EMPLOYEE individuals in Figure 5.1 are checked), reports any disclosure to the DBA and records them in the UKC and the Knowledge Set.

Population Decomposition in the Generalization Plane:

The decompose command in Figure 5.2 partitions the population of systems programmers in Figure 4.5 into three populations; systems programmers with B.S., M.S. and Ph.D. The first part of the decompose command describes the

modifications to the conceptual schema and the second part gives information about each newly created population.

The decompose command initiates the following operations.

(1) Modifications to the conceptual schema are made. Some consistency requirements such as all G-properties (see Section 4.2.3) of a newly created population exist in its subpopulations in the generalization hierarchy, etc., are satisfied.

(2) Properties of each newly created population and user's knowledge are entered into the protection data.

(3) The CMM using the QAS decides about the inferred knowledge.

Cluster Deletion of a Population:

The delete cluster command below removes the cluster consisting of the populations ASSIGNMENT-IN-DATABASE-PROJECT and ASSIGNMENT-IN-RELIABILITY-PROJECT in Figure 4.5.

```
delete cluster (ASSIGNMENT-IN-DATABASE-PROJECT,
ASSIGNMENT-IN-RELIABILITY-PROJECT);
```

The delete cluster command initiates the following operations:

(1) Clusters specified in the command are deleted from the SDB schema.

(2) G-properties of deleted populations are removed from the remaining subpopulations in the generalization

```
decompose
    parent SYSTEMS-PROGRAMMER with cluster
      (SYSTEMS-PROGRAMMER-BS, SYSTEMS-PROGRAMMER-MS,
      SYSTEMS-PROGRAMMER-PHD),
    new population SYSTEMS-PROGRAMMER-BS,
      properties SALARY(SUM,COUNT),
      individuals REL1,
      aggregate of,
      security information
        identifiability
          insertion-deletion INSERT-DELETE,
          property update SALARY(UPDATE),
        knowledge,
        global constraints ROUTINE2,
        changes processed in TRIPLETS,
        disclosure check PROGRAMMER(SALARY),
    new population SYSTEMS-PROGRAMMER-MS,
            .
            .
            .
end.
```

Figure 5.2. The Decompose Command

```
create property SALARY (SUM, COUNT),
    population SECRETARY,
    individuals INDVS,
    security information
      identifiability
        property update SALARY(UPDT),
      disclosure check EMPLOYEE(SALARY),
end.
```

Figure 5.3. The Create Property Command

hierarchy.

(3) The PDC is modified. Global constraints in other populations involving deleted populations are deleted. All UKCs are updated.

Insertion and Deletion of G-properties:

For consistency reasons, inserted properties in a population can only be G-properties and, thus, should be the property of all its sub-populations in the generalization hierarchy. The create property command in Figure 5.3 adds the property SALARY to the population SECRETARY. Notice that the create property command attaches the same statistical query types to each sub-population. If this is not desirable, the change command (described below) may be used to make modifications.

In addition to the SDB data model and protection data modifications, the QAS is also executed to decide further disclosures (similar to the create population or the decompose commands).

For consistency, only G-properties are deleted from all sub-populations having them. An example may be

```
delete attribute SALARY,
    population EMPLOYEE;
```

Similar to the delete cluster command, conceptual schema and protection data are updated as a result of the delete

attribute command.

Changing the User Group's Knowledge in a Knowledge Set:

General rules or explicit facts may be inserted or deleted using commands described below.

(1) **add general rules** INFERS
disclosure check EMPLOYEE(SALARY), ENGINEER(DEGREE);

(2) **add facts** FACT,
disclosure check EMPLOYEE(SALARY),
individual JOHN-DOE(DEGREE);

(3) **delete general rules** INFERS;

(4) **delete facts** FACT;

INFERS in (1) and (3) is a binary relation with tuples (userid,inf-rule), where inf-rule describes an general rule. Similarly FACT in (2) and (4) is a binary relation with tuples (userid,fcts), where fcts describes an explicit fact. In (2), the DEGREE of JOHN DOE is checked for disclosure as well as the SALARY properties of all individuals in EMPLOYEE population.

Changing Constraints and Other Security-Related Information:

The add constraint and the delete constraint commands may be used to insert or remove constraints from populations, for example,

add constraint (ROUTINE33,ASSIGNMENT(COUNT));

```
delete constraint (R31,EMPLOYEE(PAY-RATE));
```

ROUTINE33 and R31 are the names of modules which belong to the CEC. Notice that all these constraints are inserted into the PDC of related population. Constraints may also be inserted into the UKC of specific user groups similarly. Finally, the change command may be used to modify the security-related information in the PDCs or UKCs. An example may be

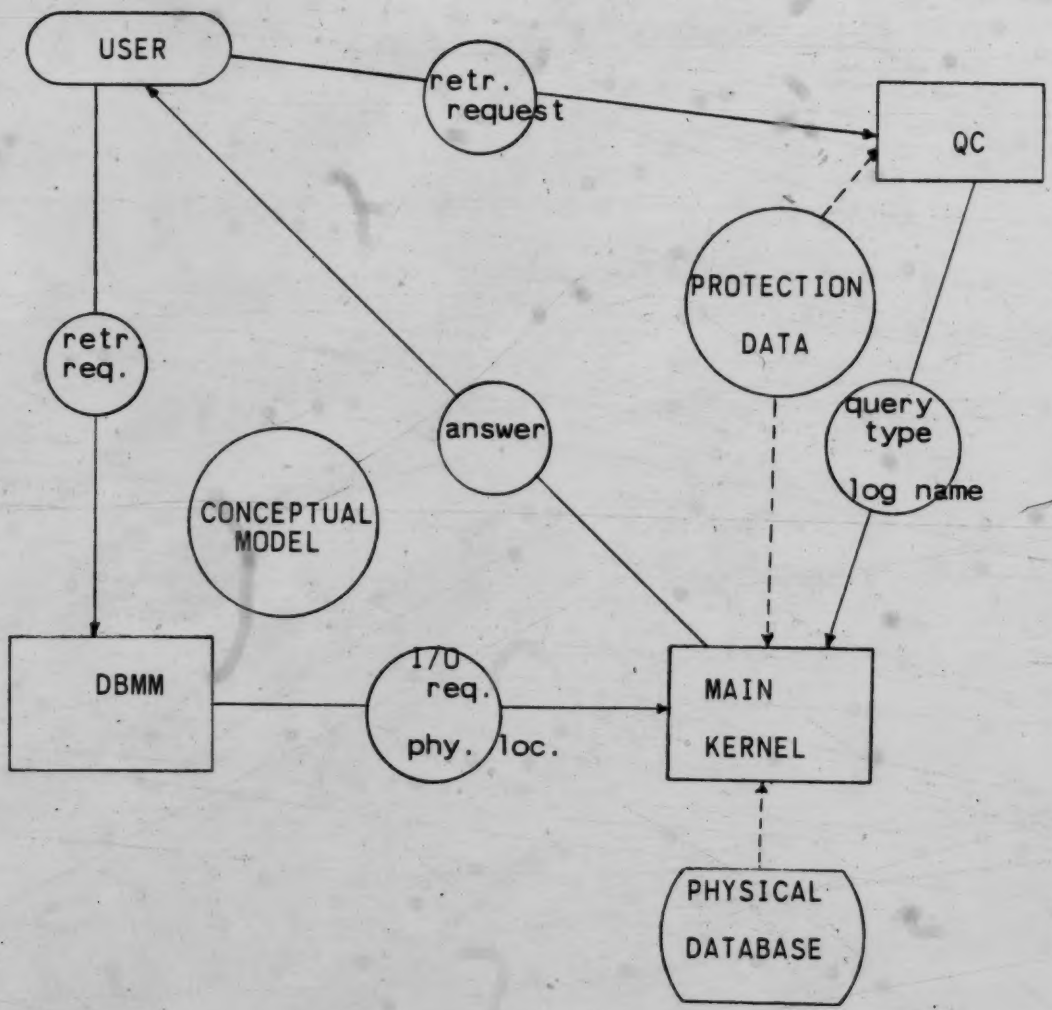
```
change population EMPLOYEE,
        identifiability INS-DEL,
        statistical query types SALARY(SUM,MEAN),
        changes processed in PAIRS,
end.
```

5.3 Processing Queries and Commands

In this section we describe how statistical queries and security-related commands are processed. Note that the DBA may be equipped with high-level data manipulation operators (such as join, project or select of relational model). However, queries involving these operators may be processed as described in [Downs and Popek, 1977] and will not be discussed here.

Statistical Queries:

Figure 5.4 shows the steps in processing statistical queries. The queries may be characteristic-specified or population-specified. The QC parses the statistical



Rectangles: Modules

Circles: Data

Dashed lines: Data flow

Solid lines: Execution flow

Figure 5.4. Retrieval Operation for Statistical Queries

retrieval request and retains the request type (SUM, MAX, MEAN, etc.) and the logical name for population-specified queries.

For characteristic-specified statistical queries, whether the specified characteristics describe an existing population in the D-A model or not is checked. if so, the name of the population and the query type are retained, otherwise the query is rejected.

The QC then passes to the main kernel the type of the query, the logical population name and the user's identification supplied by the operating system. At the same time the request is also sent to the DBMM. The DBMM functions like a normal database management system: access paths and access methods are decided, performance and other statistics are recorded, etc.. However the DBMM is not allowed direct access to the database, and it prepares and passes to the Main Kernel a read command specifying the physical locations that are to be accessed.

Accesses to the physical database are only done by the certified Main Kernel. The Main Kernel consists of I/O and authorization modules and the CEC. Before allowing access to the information in the database, the Main Kernel checks the protection data's authorization information to see if the user is allowed to access the particular population. The request from the DBMM includes physical location parameters

and logical individual names. (Both of these information are verified correct by the Main Kernel after the accesses are made). Since the protection data identifies individuals and populations by their names in logical name tables, the Main Kernel first accesses physical property values, matches them with the logical individual names and their related property specified by the request from the QC and verifies the correctness of the physically retrieved data.

Once the data is retrieved, the CEC in the Main Kernel enforces the security constraints and returns the result to the user directly.

In the above retrieval process, all the security unrelated functions of the database are separated from security related functions. Although the DBMM makes optimization decisions (access path finding, etc.), it has no authority to change the conceptual model, the protection data or the physical database. In fact, the DBMM does not know the existence of the protection data. Two security-related modules, the QC and the Main Kernel, are certified and, hence, assured of secure operation. The protection data is accessed only by kernel modules during the statistical retrieval operation.

User (or DBA) -Requested Insertion, Deletion or updates of individuals:

Figure 5.5 shows the execution of a change operation. Up to the Main Kernel, the processes are similar to the execution of statistical queries. If the change is the insertion of an individual, the CEC checks (say, by scanning the change sequence) to verify that this is the only insertion of the individual. The change (say, insertion of individual x into an SA-population) is recorded into the related change sequence of the user group requesting the change. Then for each user group u and for each property value of x and other individuals (if any) to be processed with x, the CEC sends the proper questions (see Section 5.1.1) to the QAS (with the temporary inclusion of the knowledge of processing x and the other individuals into the Knowledge Set of u). If the QAS confirms that there is no disclosure then the change is processed for the user group u (i.e. it is revealed to the users in user group u and recorded into the related change sequence of u). Otherwise the change is not processed and the change information is stored as a SA-constraint in the UKC of the user group u. The constraint and the possible disclosure are also reported to the DBA.

Normally, the change immediately takes place in the physical database and the conceptual model. However, some security constraints may exist requiring the change not to

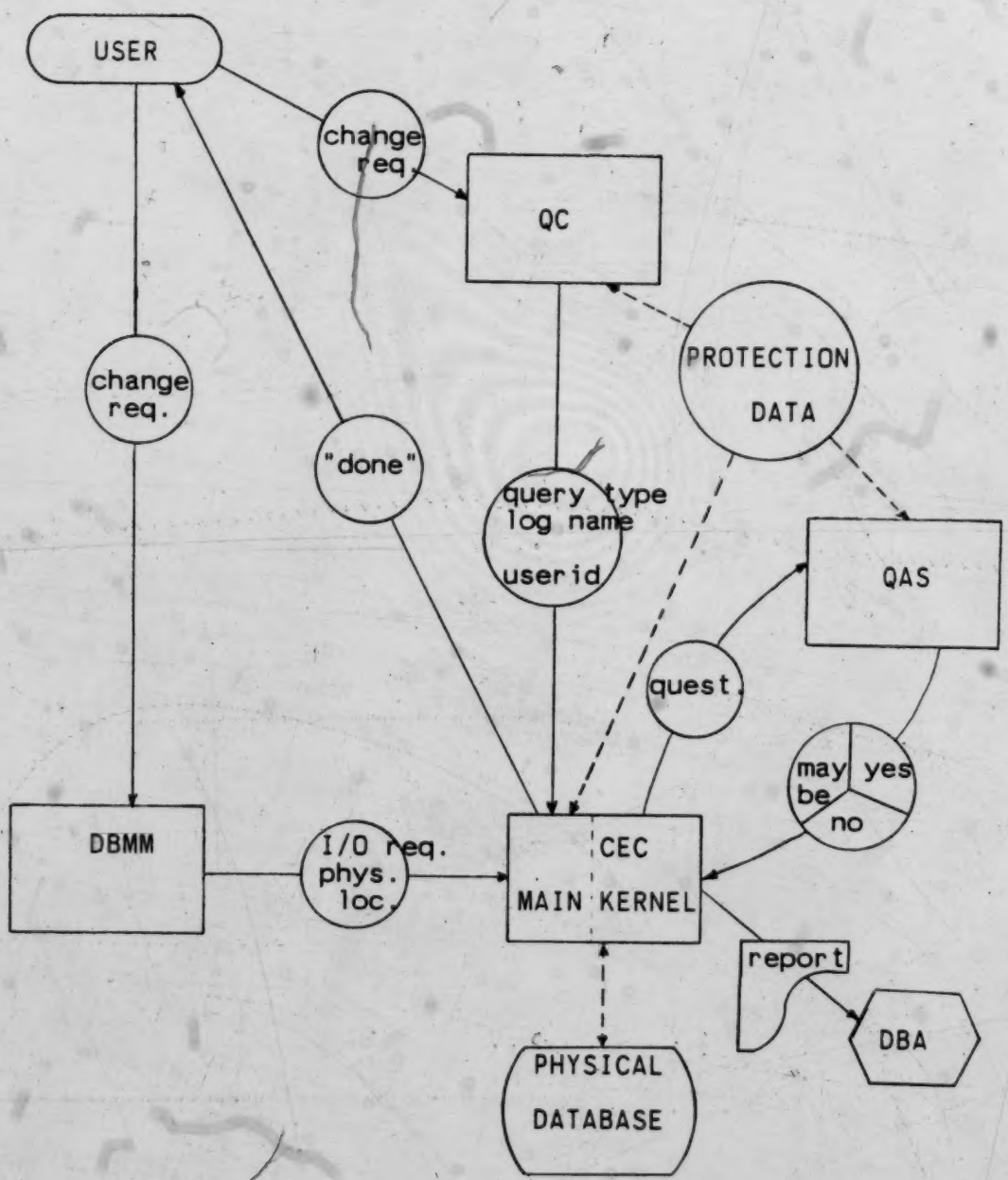


Figure 5.5. User or DBA-Requested change operation

be reflected into the answers of the statistical queries.

DBA-Requested Conceptual Model Changes:

In Section 5.2.3, six conceptual model modification commands have been identified: create population, decompose population, delete cluster, change population, create property and delete property commands. The certified CMM controls the execution of these commands. The general functions of the CMM are given below.

(a) The CMM keeps the conceptual model consistent during changes, and the security-related rules are satisfied.

(b) The CMM prepares the input to the QAS and the output to the DBA about the disclosures due to the proposed conceptual model changes. The populations and their properties specified by the disclosure check keyword in the command identify the scope of search for disclosure.

(c) The CMM updates the protection data as specified by the command. New individual objects are recorded into the logical name tables. The CMM also enters disclosures found by the QAS into the UKC and the Knowledge Set of the user group.

(d) The CMM delegates the task of making changes in the conceptual schema, external/conceptual mappings, conceptual/physical mappings and the data dictionary to the DBMM since they are not security-related.

(e) Physical database changes requested by the DBA (such as new individuals, new attributes, etc.) are passed on to

the Main Kernel by the CMM.

Figure 5.6 shows a DBA-requested conceptual model change operation. The QC parses the query, converts the explicit facts and general rules into an internal form and sends the request type, changes and other security-related information to the CMM. The CMM makes consistency checks, detects further disclosure by communicating with the QAS and reports it to the DBA, delegates the task of making conceptual model changes to the DBMM, updates the protection data and sends the physical data changes to the Main Kernel and the DBMM.

The DBMM decides about access paths and access methods, prepares necessary locking information of the physical database, etc., and sends the I/O request together with physical locations to the Main Kernel. The Main Kernel retrieves property values of individuals, compares individuals for correct logical-physical mapping, ensures correctness of the operation, makes the changes and informs the DBA about the completion of the operation.

DBA-Requested Security-Related Commands:

Section 5.2.3 lists six security-related commands, namely, addition and deletion of general rules, explicit facts and security constraints. All of these commands can only be used by the DBA and effect only the protection data. Executions of these commands are similar to the

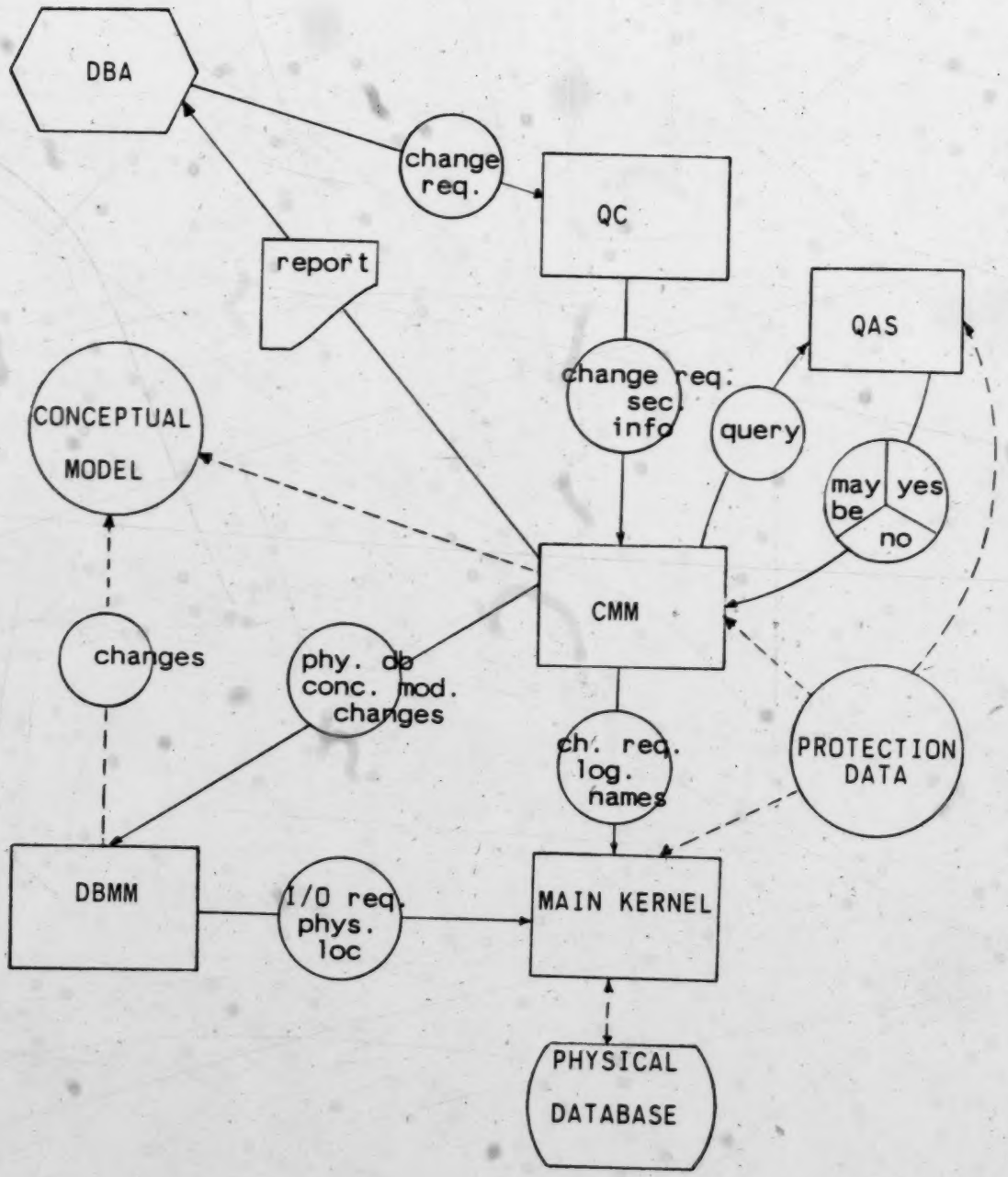


Figure 5.6. DBA-Requested Conceptual Model Change Operation

DBA-requested change operations except that there are no physical database changes and thus the DBMM and the Main Kernel are not involved. Figure 5.7 shows the execution of a DBA-requested security-related command.

5.4 Discussion

The SDB design proposed in this chapter increases the effectiveness of the protection scheme and the richness of the SDB. However efficiency of protection may degrade for two reasons.

(1) The QAS confirms the security of information by showing that the information is not deducible. As it has been pointed out to us [Schubert, 1979], the best way to show that something is not provable may not be to try to prove it. Clearly, confirming the security of secure information may become inefficient if the whole search space is to be searched. Improvement regarding this problem is possible when (a) the search space is very small, (b) irrelevant data or general rules are avoided during the deductive process, and (c) the deductive search is aided by semantic information. Another approach which may improve the efficiency of the SDB is to compile the intensional part of each user group's Knowledge Set once, using a suitably designed interactive theorem prover [Reiter, 1979]. One of the advantages of this approach is that it eliminates the need for a theorem prover at query evaluation time.

(2) Storage requirements of the protection data may be

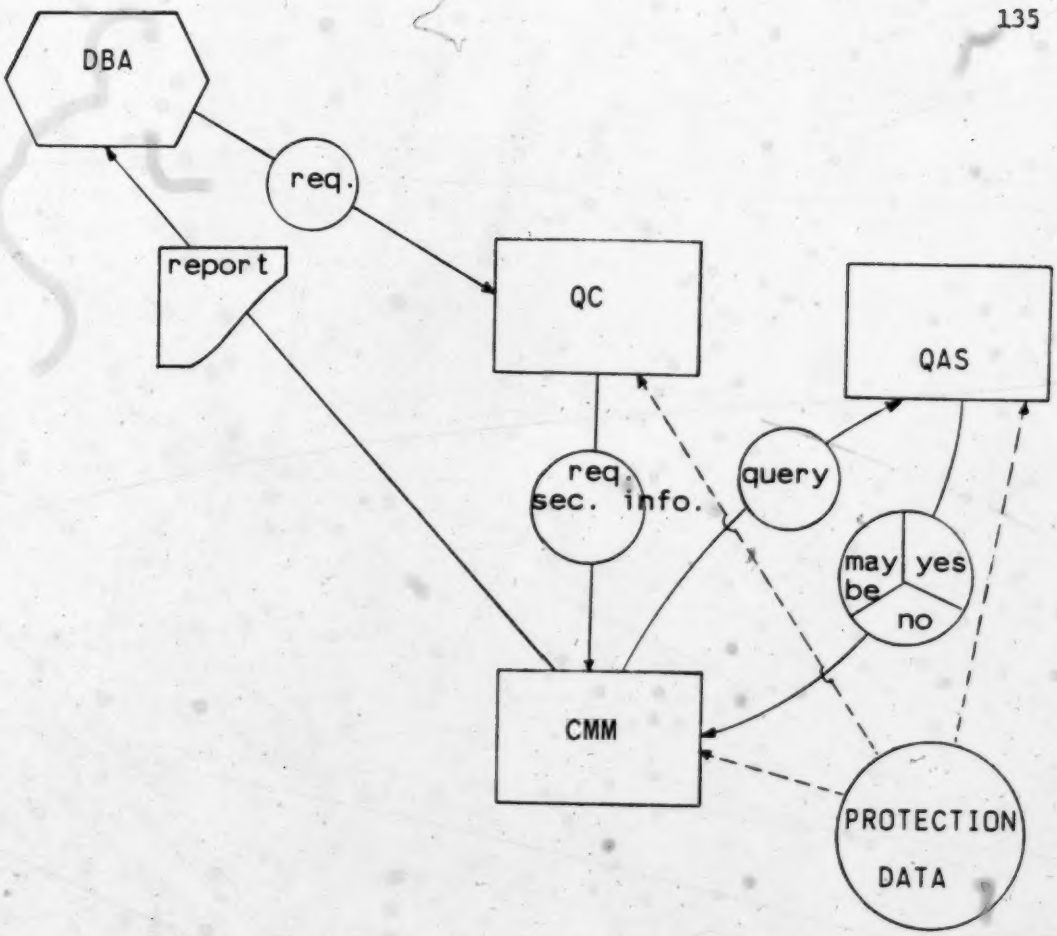


Figure 5.7. Execution of a DBA-Requested, Security-Related Command

too high when there are several Knowledge Sets with duplicate information. A solution may be to keep a single Knowledge Set with additional information as to "who" knows "what".

CHAPTER 6

CONCLUSIONS

In this thesis the security of statistical databases is investigated in the context of a statistical database design. The importance of semantic meaningfulness of users' queries is stressed. It is argued that this will enhance the security by not permitting malicious users to form meaningless queries in order to use their responses in combinatorial formulas for compromise. A natural extension of this argument and others led to the usage of semantic, redundant and structural conceptual data models in the design of a statistical database. New results involving a comprehensive secure SDB design have been described in this thesis.

The partitioning model is discussed in order to be used as a tool in the SDB design. Primitive change operations are allowed in the model, and the conditions are derived to prevent compromise. Variations of the partitioning model which use rounding, data perturbation and both are introduced to remove some of the assumptions made in the partitioning model.

Within the context of a formal framework, an SDB design using security constraints at the conceptual data model level is proposed. Three different structural, semantic and redundant data models are investigated and the D-A model [Smith and Smith, 1977] is chosen as a conceptual data model of the SDB. The population concept is utilized to identify semantically well-defined objects about which statistical information is revealed to users. For this purpose, a simple construct called the Population Definition Construct (PDC) is introduced for each population in the conceptual model.

It is argued that, for complete protection, users' additional knowledge should be maintained and kept up-to-date. Users' additional knowledge may take the form of general rules and explicit facts. The SDB design proposed herein maintains users' knowledge of only protected property values of individuals in the SDB, and this is implemented using a simple construct called the User Knowledge Construct (UKC).

In order to keep the PDCs and UKCs up-to-date, to enforce the security constraints and to help the DBA in security-related decision problems, the constraint enforcer and checker (CEC) is proposed. The CEC, UKCs and PDCs comprise the Statistical Security Management Facility (SSMF). Implementation issues of the SSMF are briefly discussed.

A novel property of the SDB design in Chapter 4 is that it can be "added on" to an existing general-purpose database system without major modifications to its DBMS. This feature may be useful if the general-purpose database has confidential information about which statistical queries are permitted. In such a case, the SSMF is added to the existing DBMS and security constraints are enforced for each statistical user query.

Different types of inferences by users are identified, and possible security constraints for different types of statistical queries are investigated. It is demonstrated that, usually simple security constraints can be defined to protect the SDB from compromise.

Extensions to the SDB design are described in order to increase effectiveness of the protection and the richness of the SDB. A question-Answering System with deductive inference mechanisms is proposed for deciding the inferred knowledge. For each user group, a Knowledge Set which keeps the user group's additional knowledge of general rules and explicit facts relevant to the application is proposed. A set of security-related commands for changes in the conceptual model are proposed (for the D-A model) in order to give the SDB the capability to reflect the changes in the real world. It is argued that a security kernel architecture will enhance the security of the SDB. Finally, an SDB design

is discussed which includes these extensions. Problems with the efficiency of the design are summarized.

There are several avenues of research related with our approach of SDB design. One direction of research is to find some at least semi-automated security measures to help the DBA assess how secure the SDB is. In this thesis, using the information graph, two very simple security measures are defined. However, both of these measures are very crude and apply to only SUM and COUNT queries. Some, maybe probabilistic, security measures are needed for all types of statistical queries. Another research area is the deductive components of the QAS. Clearly more research is needed there to make the SDB design in Chapter 5 efficient. Finally, the proposed SDB design has yet to be implemented and tested in a real-world application. This may shed new light on the problem of secure SDB design.

The SDB design in this thesis was motivated by the premise of supplying the SDB users with only what they need, i.e., as a prerequisite, a semantically meaningful information. It seems to the author that further categorization of the needs of users may dictate a hierarchically designed SDB with totally different atomic information units, security constraints and constructs at each level of the hierarchy. Consider, for example, a large company with its executives, managers and engineers, etc.. Clearly the statistical information needed by engineers for

research purposes is quite different than the statistical information needed by its executives for decision-making purposes.

In this thesis, output and data perturbation techniques have been used only to remove some deficiencies of the partitioning model. These protection techniques and sampling may be very useful in those cases where SDB users' needs are very diverse or the implementation of a conceptual model and the SSMF are not feasible. There are promising research results about the protection techniques of data perturbation and random sampling [Beck, 1979; Denning, 1979], however, our understanding about the efficiency, effectiveness and usefulness of these techniques is yet to be furthered. Another research direction may be to investigate auditing. Clearly, one measure of the SDB security problem is the number of queries that the user has asked, i.e. the more queries, the more likely danger of compromise. Fast auditing algorithms under different SDB models may be yet another research direction.

BIBLIOGRAPHY

Achugbue J. D., Chin F. Y., "The Effectiveness of Output Modification by Rounding for Protection of Statistical Databases" INFOR, 17, 3, 1979, pp. 209-218.

ACM, "Executive Guide to Computer Security", 1974.

Aho A. V., Hopcroft J. E., Ullman J. D., The Design and Analysis of Computer Algorithms, Addison Wesley, 1976.

Beck L. L., "A Security Mechanism for Statistical Databases", Dept. of Comp. Science, Southern Methodist Univ., 1979.

Bledsoe W. W., Bruell P., "A Man-Machine Theorem-Proving System", Artificial Intelligence, 5, 1, 1974, pp. 51-72.

Boruch R. F., "Maintaining Confidentiality in Educational Research: A Systematic Analysis", Amer. Psychologist 26, 1971, pp. 413-430.

Chang C. L., Lee R. C. T., "Symbolic Logic and Mechanical Theorem Proving", Academic Press, New York, 1973.

Chen P. P-S., "The Entity-Relationship Model-Toward a Unified View of Data", ACM Trans. Database Systems, 1, 1, 1976, pp. 1-36.

Chin F. Y.,"Security in Statistical Databases for Queries with Small Counts", ACM Trans. Database Syst. 3, 1, 1978, pp. 92-104.

Chin F. Y., Özsoyoğlu G.,"Security in Statistical Databases for Sum and Count Queries", Information Privacy, 1, 4, 1979, pp. 148-153.

Chin F. Y., Özsoyoğlu G., "Security in Partitioned Dynamic Statistical Databases", Proc., IEEE COMPSAC Conf., 1979, pp. 594-601.

Chin F. Y., Özsoyoğlu G.,"Statistical Database Design", Dept. of Computing Sciences, Univ. of Alberta, 1979. (Submitted to ACM TODS).

Chin F. Y., Özsoyoğlu G., "Security of Statistical Databases", Dept. of Computing Sciences, Univ. of Alberta, 1979.

Codd E. F., "A relational model for large shared data banks", CACM 13, 6, 1970, pp. 377-387.

Codd E. F., " Recent investigations in relational data base

systems", Information Processing 74, North-Holland Pub. Co., Amsterdam, 1974, pp. 1017-1021.

Codd E. F., "Extending the data Base Relational Model to Capture more Meaning", ACM Trans. Database Syst. 4, 4, 1979, pp. 397-434.

Dalenius T. and Reiss S. P., "Data Swapping-A Technique for Disclosure Control", Confidentiality in Surveys, Rep. no. 31, Dept. Stat., Univ. Stockholm, 1978.

Date C. J., "An Introduction to Database Systems", Addison Wesley, 1977.

Davida G., Linton D., Szelag G., Wells D., "Security of Statistical Databases", TR-CS-76-14, Dept. of EECS, Univ. of Wisconsin, Milwaukee, 1976.

Davida G., Rocheleau R., "Compromising a Database Using MEAN Queries of Variable Length", TR-CS-77-2, Univ. of Wisconsin, Milwaukee, 1976.

Davida G. I., Wells D. L., Kam J. B., "Security and Privacy", Proc., IEEE COMPSAC Conf., 1978, pp. 194-203.

DeMillo R. A., Dobkin D.,"Recent Progress in Secure Computation", Proc., IEEE COMPSAC Conf., 1978.

DeMillo R., Dobkin D., Lipton R. J.,"Even Databases That Lie Can Be Compromised", IEEE Trans. Software Engineering, SE-4, 1, 1978, pp. 73-75.

Denning D. E., "Are Statistical Databases Secure", Proc. AFIPS NCC, 1978, pp. 525-530.

Denning D. E., "Secure Statistical Databases with Random Sample Queries", Computer Sciences, Purdue Univ., 1979.

Denning D. E., Denning P. J., "Data Security", ACM Computing Surveys, 11, 3, 1979, pp. 227-250.

Denning D. E., Schlorer J., "A Fast Procedure for Finding a Tracker in a Statistical Database", ACM Trans. Database Syst., 5, 1, 1980.

Denning D. E., Denning P. J., Schwartz M. D., "The Tracker: A Threat to Statistical Database Security", ACM Trans. Database Syst. 4, 1, 1979, pp. 76-96.

Dobkin D., Jones A. K., Lipton R. J., "Secure Databases: Protection Against User Inference", ACM Trans. Database Syst., 4, 1, 1979, pp. 97-106.

Dobkin D., Lipton R. J., Reiss S. P.,"Aspects of the Database Security Problem", Proceedings on a Conference on Theoretical Computer Science, Waterloo, Canada, 1977.

Downs D., Popek G. J., "A Kernel Design for a Secure Data Base Management System", Proc., 3rd VLDB Conf., 1977, pp. 507-514.

Downs D., Popek G. J., "Data Base Management Systems Security and INGRES", Proc., 5th VLDB Conf., 1979, pp. 280-290.

Fellegi I. P.,"On the Question of Statistical Confidentiality", J. Amer. Statist. Assoc. 67, 337, 1972, pp. 7-18.

Fellegi I. P., Phillips J. L.,"Statistical Confidentiality: Some Theory and Applications to Data Dissemination", Annals of Econ. Soc'l Measurement, 3, 2, 1972, pp. 399-409.

Fernandez E. B., Summers R. C., Coleman C. D.,"An Authorization Model for a Shared Database", Proc., ACM SIGMOD Conf., 1975.

Garey M. R., Johnson D. S., Computers and Tractability. W. H. Freeman and Company, San Fransisco, 1979.

Green C. C., "Theorem Proving by Resolution as a Basis for Question-Answering Systems", in Machine Intelligence 4, Ed.s Meltzer and Michie, Edinburgh Univ. Press, Edinburgh, 1969, pp. 183-205.

Griffits P. P., Wade B. W., "An Authorization Mechanism for a Relational Database System", ACM Trans. Database Syst., 1, 3, 1976, pp. 242-255.

Haq M.,"Insuring Individual's Privacy from Statistical Database Users", Proc. AFIPS NCC, Vol. 14, AFIPS Press, 1975, pp. 941-946.

Hartson H. A., Hsiao D. K.,"A Semantic Model for Database Protection Languages", Proc., 2nd VLDB Conf., pp. 27-42, 1976.

Hammer M. M., McLeod D. J.,"Semantic Integrity in a Relational Database System", Proc., 1st VLDB Conf., 1975, pp. 25-47.

Hansen M. H.,"Insuring Confidentiality of Individual Records in Data Storage and Retrieval for Statistical Purposes", Proc. AFIPS FJCC, 39, 1971.

Hoffman L. J., Modern Methods for Computer Security and

Privacy, Prentice-Hall, 1977.

Hoffman L. J., Miller W. F.,"Getting a Personel Dossier from a Statistical Data Bank", Datamation 16, 5, 1970.

Hsiao D. K., Kerr D. S., Madnick S. E., "Privacy and Security of Data Communications and Databases", Proc., 4th VLDB Conf., 1978.

Kam J. B., Ullman J. D.,"A Model of Statistical Databases and Their Security", ACM Trans. Database Syst. 2, 1, 1977, pp. 1-10.

Kellogg C., Klahr P., Travis L., "A Deductive Capability for Data Management", Proc., 2nd VLDB Conf., 1976, pp. 181-196.

Kernighan B. W., Lin S., " An Efficient Heuristic Procedure for Partitioning Graphs", The Bell Syst. Tech. Journal, 1970, pp. 291-307.

Lukes, J. A., "Efficient Algorithm for the Partitioning of Trees", IBM J. Res. Develop., 1974, pp. 217-223.

Kerschberg L., Klug A., Tsichritzis D., "A Taxonomy of Data Models", Proc. 2nd VLDB Conf., 1976, pp. 43-63.

Madnick S. E., Computer Security, Academic Press, New York, 1979.

Minker J., Fishman D. H., McSkimin J. R., "The Q* Algorithm--A Search Strategy for a Deductive Question-Answering System", Artificial Intelligence, 4, 3, 1973, pp. 225-243.

Minker J., "Search Strategy and a Selection Function for an Inferential Relational System", ACM Trans. Database Syst., 3, 1, 1978, pp. 1-32.

Mood A. M, Graybill F. A., Boes D. C., Introduction to the Theory of Statistics, McGraw-Hill, New York, 1974.

Nargundkar M. S., Saveland W., "Random Rounding to Prevent Statistical Disclosure", Proc. Amer. Stat. Assoc., Soc. Stat. Sec., 1972, pp. 382-385.

Nielsen N. R., Ruder B., Brandin D. H., "Effective Safeguards for Computer System Security", Proc. AFIPS NCC, Vol. 45, AFIPS Press, 1976, pp. 75-84.

Nijssen G. M. (Ed.), IFIP Working Conference on Modelling in Data Base Management Systems, Proceedings, North-Holland, 1976.

Nilsson N. J., "Problem-Solving Methods in Artificial Intelligence", McGraw-Hill, 1971.

Özsoyoğlu G., Chin F. Y., "Enhancing the Security of Statistical Databases with a Question-Answering System and a Kernel Design", Tech. Report, Dept. of Computing Science, Univ. of Alberta, 1979.

Privacy Act of 1974, Title 5, United States Code, Section 552a (Public Law 93-579), 1974.

Parker D. B. Crime by Computer Scribner's, New York, 1976.

Reiss S. P.,"Security in Databases: A Combinatorial Study", JACM, 26, 1, 1979.

Reiss S. P., "Medians and Database Security", Foundations of Secure Computations, Academic Press, 1978, pp. 57-92.

Reiter R., "Deductive Question-Answering in Relational Databases", in Logic and Databases, Ed.s H. Gallaire and J. Minker, Plenum Press, N.Y., 1978.

Robinson J. A., "A Machine-Oriented Logic Based on the Resolution Principle", JACM, 12, 1, 1965, pp. 23-41.

Schlorer J.,"Identification and Retrieval of Personel Records from a Statistical Databank", Methods of Info. in Medicine, 14, 1, 1975, pp. 7-15.

Schlorer J.,"Confidentiality of Statistical Records: A Threat Monitoring Scheme for On-line dialoque", Methods of Info. in Medicine, 15, 1, 1976, pp. 36-42.

Schlorer J., "Union Tracker and Open Statistical Databases", TB-IMSD 1/78, Inst. Med. Statist. Dok., Univ. Giessen, 1978.

Schlorer J., "Security of Statistical Databases: Multidimensional Transformation", TB-IMSD 2/78, Inst. Med. Statist. Dok., Univ. Giessen, 1979.

Schwartz M. D., Denning D. E., Denning P. J., "Linear Queries in Statistical Databases", ACM Trans. Database Syst., 4, 2, 1979, pp. 156-167.

Schubert, L. K. Private Communication.

Shankar K. S., "The Total Computer Security Problem: an Overview", Computer, 10, 6, 1977, pp. 50-73.

Smith J. M., Smith D. C. P.,"Database Abstractions: Aggregation", CACM, 20, 6, 1977.

Smith J. M., Smith D. C. P.,"Database Abstractions: Aggregation and Generalization", ACM Trans. Database Syst., 2, 2, 1977, pp. 105-133.

Weber H., "A Semantic Model of Integrity Constraints on a Relational Database", In "Modelling in Data Base Management Systems", Ed. G. M. Nijssen, North-Holland Publishing Company, New York, 1976.

Winograd T. A., "A Procedural Model of Language understanding", in "Computer Models of Thought and Language", Ed.s. Schank R. C. and Colby C. M., W. H. Freeman and Company, San Fransisco, 1973.

Yu C. T., Chin F. Y.,"A Study on the Protection of Statistical Databases", Proc., ACM SIGMOD Conf., 1977.

Appendix A

Proof of Lemma in Section 3.2.3: Let $b'=(b-1)/2$ and $b''=1+(b-1)/2$. We will first show that $p_R(j)=p_R(b-j)$, $1\leq j\leq(b-1)/2$, where p_R is the probability function for the r.v. R. Since the v_i's and T are symmetrically distributed, from (2) and (3) in Section 3.2.3, S is also symmetrically distributed about the mean $M_S=M_V.M_Z$, and its variance is

$$\mathrm{Var}(S) = \mathrm{Var}(T) + M_Z.\mathrm{Var}(V) + \mathrm{Var}(Z).M_V^2$$

Clearly, for $1\leq j\leq(b-1)/2$,

$$p_R(j)=\sum_i p_S(ib+j) \quad \text{and} \quad p_R(b-j)=\sum_i p_S(ib-j)^+$$

Since $M_S=kb$, for $-k\leq i\leq k$, $1\leq j\leq(b-1)/2$, we have

$$p_S((k-i)b+j) = p_S((k+i)b-j) \tag{1}$$

and for $i=-1,-2,\ldots.$ and $1\leq j\leq(b-1)/2$, we have

$$p_S(ib+j) = p_S((2k+|i|)b-j) \tag{2}$$

and

$$p_S((2k+|i|)b+j) = p_S(ib-j) \tag{3}$$

Addition of (1),(2) and (3) gives

$$\sum_i p_S(ib+j) = \sum_i p_S(ib-j)$$

or

$$p_R(j) = p_R(b-j) \tag{4}$$

+The notation $\sum_y$ denotes summation over all possible integer y values.

To show (a):

Since

$$p_W(w) = \text{Prob}(W=w) = \sum_{r=\emptyset}^{b-1} \text{Prob}(W=w|R=r).p_R(r)$$

$$= p_T(w).p_R(\emptyset) + \sum_{r=1}^{b'} p_T(w+r).p_R(r) + \sum_{r=b''}^{b-1} p_T(w+r-b).p_R(r)$$

we have

$$M_W = p_R(\emptyset).\sum_w p_T(w).w + \sum_{r=1}^{b'} p_R(r).[\sum_w w.p_T(w+r)]$$

$$+ \sum_{r=b''}^{b-1} p_R(r).[\sum_w w.p_T(w+r-b)]$$

$$= \sum_{r=1}^{b'} p_R(r).[\sum_t (t-r)p_T(t)] + \sum_{r=b''}^{b-1} p_R(r).[\sum_t (t+b-r)p_T(t)]$$

$$= \sum_{r=b''}^{b-1} (b-r)p_R(r) - \sum_{r=1}^{b'} r.p_R(r) = \sum_{r=1}^{b'} r[p_R(b-r) - p_R(r)]$$

$$= \emptyset \text{ due to (4).}$$

To show (b):

$$\text{Var}(W) = E(W^2) = \sum_w w^2.p_W(w) = \sum_{r=\emptyset}^{b'} p_R(r).[\sum_t (t-r).p_T(t)]$$

$$+ \sum_{r=b''}^{b-1} p_R(r).[\sum_t (t+b-r).p_T(t)]$$

$$= \sum_{r=\emptyset}^{b'} p_R(r).(\text{Var}(T)+r^2) + \sum_{r=b''}^{b-1} p_R(r).[\text{Var}(T)+(b-r)^2]$$

$$= \text{Var}(T) + \sum_{r=\emptyset}^{b'} r^2.p_R(r) + \sum_{r=b''}^{b-1} p_R(r).(b^2-2br+r^2)$$

$$= \text{Var}(T) + E(R^2) + b.\sum_{r=b''}^{b-1} p_R(r).(b-2r)$$

$$= \text{Var}(T) + E(R^2) - b.\sum_{r=1}^{b'}(b-2r)$$

$$= \text{Var}(T)$$

since

$E(R^2) = \sum_{r=1}^{b-1} r^2.p_R(r)$ and, using (4) we have

$$= \sum_{r=1}^{b'}(r^2+(b-r)^2).p_R(r) = b\sum_{r=1}^{b'}p_R(r).(b-2r) \qquad \#$$

Appendix B

Usage of Dummy Records in the Partitioning Model:

(1) Assume a path has two active vertices, then while inactivating the active vertices (during their deletion), one may use two dummy records, and then forces the path into a cycle by deleting the two dummy records.

(2) Assume a path has a single active vertex, then while inactivating the active vertex (during its deletion), one may introduce a new active dummy record, dr_j, and later on, when there are sufficient (e.g. t) dummy records like dr_j, they may be deleted from the partition altogether.

(3) Assume r_a and r_b are requested to be in different reachability sets. Assume also the vertex r_a is already in the information graph and vertex r_b is to be formed. If vertex r_b has a danger of being connected to vertex r_a then the system uses a dummy record dr_j to create a new path with vertices r_b and dr_j; afterwards, either by checking at each change operation or making one of the paths that r_a and r_b belongs to inactive, the system may keep the two paths separate.

Some of the updated records may run the danger of being traced by the user. When this happens, there is a possibility of an odd cycle since the information graphs of two different partitions are connected. When the usage of

dummy records are allowed, a procedure similar to (3) described above can be used to prevent the formation of odd cycles for "traceable" update operations; if two records r_a, r_b which are in the same reachability set of partition p_i, are moved to another partition p_j then they may be kept in two disconnected paths using (3) described above.